Universidade Nova de Lisboa
Faculdade de Ciências e Tecnologia
Departamento de Informática

Dissertação de Mestrado

Mestrado em Engenharia Informática

Troca de serviços e incentivos em sistemas P2P

Ricardo Jorge Cordeiro Duarte da Silva (aluno nº 25812)

1º Semestre de 2009/10

22 de Fevereiro de 2010

Universidade Nova de Lisboa
Faculdade de Ciências e Tecnologia
Departamento de Informática

Dissertação de Mestrado

Troca de serviços e incentivos em sistemas P2P

Ricardo Jorge Cordeiro Duarte da Silva (aluno nº 25812)

Orientador: Prof. Doutor José Augusto Legatheaux Martins

Trabalho apresentado no âmbito do Mestrado em Engenharia Informática, como requisito parcial para obtenção do grau de Mestre em Engenharia Informática.

1º Semestre de 2009/10
22 de Fevereiro de 2010

Agradecimentos

Em primeiro lugar gostava de agradecer ao Professor Doutor José Augusto Legatheaux Martins, orientador da dissertação, por todo o apoio e por todas as valiosas contribuições para o trabalho. Acima de tudo, obrigado pelas muitas horas de debate que me incentivaram a continuar e me ajudaram a manter no rumo certo.

Aos meus colegas e amigos, gostava de agradecer pela constante motivação e ajuda que deram. Em particular ao Vítor pelo debate de ideias de implementação, ao Nuno, Bruno e Hélio pela experiência de quem já passou pelo mesmo, oferecendo os seus concelhos e revisões do documento, ao Danilo e a todos os meus companheiros de dissertação pelo companheirismo e boa disposição durante os longos meses em que decorreu esta dissertação.

Agradeço a todos os meus familiares pelo incentivo recebido ao longo de todo este tempo. Aos meus pais, à minha avó e à minha irmã Sara, obrigado pela paciência, amor e atenção. Sem as oportunidades proporcionadas por eles não me seria possível alcançar tudo o que alcancei academicamente.

Para a minha namorada Marta, deixo um agradecimento especial, não só por toda a dedicação e amor, mas também pelo estímulo intelectual e emocional que me transformaram desde o segundo ano da licenciatura e culminam na concretização desta dissertação.

O meu profundo e sentido agradecimento a todas as pessoas que contribuíram de uma maneira ou de outra para a realização desta dissertação.

Resumo

A popularidade e viabilidade da generalização das aproximações P2P para difusão de conteúdos (ficheiros) estão dependentes da utilização de mecanismos realistas de incentivo e de partilha de custos. Por exemplo, o sucesso do sistema BitTorrent deve-se em grande medida ao seu algoritmo de partilha de recursos com combate aos *free-riders*.

No entanto, esse algoritmo não é totalmente óptimo, funciona segundo um modelo de "troca directa imediata" e ignora os custos da rede e dos ISPs. A introdução de mecanismos de incentivos ou de políticas de custos e preços, incrementando a troca de serviços entre participantes e até mesmo operadoras, parece essencial para melhorar e optimizar estes sistemas.

Nesta dissertação é estudado o algoritmo do BitTorrent, as razões do seu sucesso e diversas propostas de melhorias do mesmo e de introdução de mecanismos complementares que podem melhorar o desempenho de um grupo de utilizadores ou do conjunto dos participantes na mesma rede de partilha de conteúdos.

Parte dessas propostas são analisadas experimentalmente e o resultado da sua utilização é avaliado e comparado. Conclui-se através desse estudo que um algoritmo que dê prioridade aos participantes que têm uma boa probabilidade de permanecerem na rede a fornecer serviço após a descarga dos ficheiros é mais eficaz, quer para esses participantes, quer para o conjunto da rede, representando assim um mecanismo com potencial grande interesse.

Esta dissertação inclui também uma discussão das condições em que tal algoritmo poderia ser usado com probabilidade de algum sucesso num contexto real.

Palavras-chave: Sistemas P2P, BitTorrent, Algoritmos distribuídos, Troca de incentivos, Combate ao *free-riding*.

Abstract

The popularity and viability of P2P approaches for content distribution (files) depend of the use of real incentive methods and cost sharing. For instance, the success of the BitTorrent system comes greatly from its content sharing algorithm that can fight free-riding.

Yet, this algorithm isn't totally optimal, and works under a "tit-for-tat"model that ignores the network and ISP's costs. Introducing methods of incentives or policies that manage costs and prices, allowing the trading of services between peers and even providers, seems to be essential to improve and optimize these systems.

In this thesis the BitTorrent algorithm and the reasons for its success are studied, as well as some improvement and complementary mechanisms proposals, which may improve performance for a given group of users, or for all the users of a content share network.

Some of those proposals are analyzed through simulation, and its results evaluated and compared. This study concludes that an algorithm that privileges users that have good probability of remaining connected offering service after the download of the files is more effective, not only for those users, but also for the entire swarm, thus becoming a potentially interesting mechanism.

This thesis also includes a discussion about the plausible real context deployment conditions required for this algorithm to be used.

Keywords: P2P Systems, BitTorrent, Distributed algorithms, Exchange of incentives, fighting free-riding

Conteúdo

1 Introdução **1**
1.1 Contexto e âmbito do trabalho 1
1.2 Organização do documento 2

2 O protocolo BitTorrent **3**
2.1 Publicar conteúdo 3
2.2 O Tracker 4
2.3 A partilha de ficheiros 4
2.4 Selecção de blocos 5
2.4.1 Momento inicial 6
2.4.2 Momento final 6
2.5 O algoritmo de Choking 6
2.6 Razões para o sucesso 9
2.6.1 Facilidade de utilização 9
2.6.2 Robustez 10
2.6.3 Comunidades 11
2.7 Análises 12
2.8 Em síntese 13

3 Optimização e evoluções do algoritmo original **15**
3.1 BitTyrant 15
3.2 BitMax 16
3.3 Biased Neighbor Selection 17
3.4 TRIBLER 18
3.5 2Fast 20
3.6 PACE: Peer-Assisted Content Exchange 22
3.7 PropShare 23
3.8 FairTorrent 25
3.9 Em síntese 27

4 O ambiente de experimentação **29**
4.1 Simuladores, emuladores e a Internet 29
4.2 O simulador escolhido 31
4.3 Melhoramentos introduzidos no simulador 31
4.4 Funcionamento do simulador 35
4.5 Soluções e alterações ao BitTorrent que foram implementadas 38
4.5.1 Protocolo 2Fast 38
4.5.2 Alternativa testada: Helpers colectivos 40
4.5.3 Protocolo FairTorrent 41

4.5.4 Alternativa testada: Trusted Peers 42

5 Estudos comparativos 45
5.1 Introdução 45
5.2 Protocolos testados 45
5.3 Cenários de testes 46
5.4 Resultados dos testes 48
5.4.1 Protocolo BitTorrent 50
5.4.2 Protocolo 2Fast 52
5.4.3 Helpers colectivos 56
5.4.4 Protocolo FairTorrent 58
5.4.5 Trusted Peers 59
5.5 Conclusões 62

6 Análise de viabilidade da utilização da proposta: Trusted Peers 65
6.1 Comportamento dos utilizadores 65
6.2 Política de incentivos 66
6.3 Implementação 66
6.4 Conclusões 68

7 Resumo, conclusões e trabalho futuro 71

Lista de Figuras

2.1 Exemplo ilustrativo da tendência em manter ligações com utilizadores com largura de banda igual ou superior, no algoritmo de *Choking*. À esquerda a situação inicial, à direita a situação após *optimistic unchoke*. 8

3.1 Representação de uma rede baseada no protocolo 2Fast. Retirado de [13]. 21
3.2 Cenários possíveis na óptica de interesse para o participante *i*. Estão ordenados de forma decrescente de interesse para *i*. As setas representam interesse. Retirado de [9]. 25

4.1 Interface do simulador GPS 32
4.2 Interface observável durante uma simulação no simulador GPS 36

Lista de Tabelas

5.1 Resumo dos cenários de teste 49
5.2 Tempos de transferência dos cenários 3 e 5 50
5.3 Rácios e tempos médios de transferência dos cenários 1, 1+, 3 e 3+ 51
5.4 Tempos de transferência dos cenários 1, 2 e 3 52
5.5 Rácios de cada classe de participantes nos cenários 1, 2 e 3 53
5.6 Tempos de transferência dos cenários 1+, 2+ e 3+ 54
5.7 Rácios de cada classe de participantes nos cenários 1+, 2+ e 3+ 55
5.8 Tempos de transferência do cenário 4, com os tempos dos cenários 1 e 3 para controlo 56
5.9 Rácios de cada classe de participantes nos cenários 4, com os rácios dos cenários 1 e 3 para controlo 57
5.10 Tempos de transferência do cenário 3 com e sem protocolo FairTorrent 58
5.11 Rácio dos utilizadores e número de blocos oferecido pelas sementes no cenário 3, com e sem protocolo FairTorrent 59
5.12 Tempos de transferência do cenário 6, com controlo feito pelo cenário 3 com e sem protocolo FairTorrent 60
5.13 Rácio dos utilizadores e número de blocos oferecido pelas sementes no cenário 6, com controlo feito pelo cenário 3 com e sem protocolo FairTorrent 61

1. Introdução

Hoje em dia uma das maiores formas de difusão de conteúdo (e.g. ficheiros) tem como base sistemas *peer-to-peer*, dos quais o protocolo BitTorrent é o melhor e mais amplamente conhecido exemplo. O sucesso que conheceu até aos dias de hoje deve-se essencialmente a dois pontos: a forma como é capaz de optimizar e acelerar a transferência de conteúdo, não só do ponto de vista singular, como também global, quando comparados com o modelo cliente-servidor, e o combate ao *free-riding*[1], obrigando a que cada participante na partilha contribua para acelerar a transferência de todos.

O algoritmo tem-se revelado na prática como adequado e tem cumprido com os objectivos para os quais foi criado, mas para enfrentar os desafios do futuro é necessária a criação de mecanismos mais realistas de incentivo à partilha de recursos e de optimização de recursos de rede. Será necessário atender ao facto de que cada vez mais os utilizadores se dividem por diversas redes e diversos ficheiros, e aproveitar este pressuposto para traçar novos sistemas, em que a troca de recursos não se prende apenas com o mesmo ficheiro ou no mesmo espaço temporal, abrindo assim uma janela para novos conceitos e algoritmos, que introduzam melhorias de desempenho tanto a nível individual como global.

Cria-se assim a motivação para a investigação nesta área, que tem uma importância grande face ao volume de tráfego gerado e ao número de utilizadores que abrange.

1.1 Contexto e âmbito do trabalho

Nesta dissertação é apresentado e estudado em detalhe o funcionamento do algoritmo BitTorrent e as razões que fazem do mesmo uma das mais bem sucedidas formas de partilha de conteúdo em sistemas baseados em P2P.

São apresentadas várias propostas de melhoramentos do algoritmo como o BitTyrant, que usa uma estratégia alternativa com o objectivo de mostrar as fragilidades do sistema, ou o BitMax, que se baseia em condições impostas ao comportamento dos utilizadores para usar uma forma alternativa de trocas. Outras propostas apresentadas abordam o problema pelo lado dos ISPs, que têm de lidar com uma grande quantidade de tráfego gerado pelo protocolo. Existem

[1] *Free-rider* é a expressão inglesa que designa utilizadores que usam o sistema sem contribuir para o mesmo. Para facilitar a leitura utiliza-se o termo inglês.

ainda propostas que introduzem mecanismos complementares como os casos do TRIBLER e do 2Fast que introduzem novas classes de utilizadores: Collectors e Helpers, ou o sistema PACE, que introduz um sistema baseado em noções económicas de equilíbrio de mercado para regular as trocas entre participantes. Por fim são apresentados os sistemas PropShare e FairTorrent, que alteram o conceito actual de trocas, e abrem as portas à introdução de novas soluções.

Algumas destas propostas foram analisadas experimentalmente e o resultado da sua utilização foi estimado e comparado. Os testes foram realizados num ambiente de simulação.

A análise dos resultados obtidos revelou que o incentivo à criação e manutenção de sementes no sistema é crucial para aumentar o seu desempenho, o que levou à principal contribuição inovadora deste trabalho: a solução Trusted Peers. Esta solução parte de um princípio muito simples e que é o seguinte: se de alguma forma se identificarem os utilizadores que mais contribuem para a rede, permanecendo nela como sementes, e se lhes for dada prioridade nas transferências, então quer estes utilizadores, quer o conjunto da rede têm vantagens. A solução foi implementada e testada através de uma modificação do protocolo FairTorrent. Para além de demonstrar através de simulação que esta proposta produz os resultados pretendidos é discutida a viabilidade da sua utilização em ambiente real.

1.2 Organização do documento

Após esta introdução, os restantes capítulos deste documento estão organizados da seguinte forma: o protocolo BitTorrent será estudado pormenorizadamente no capítulo 2. No capítulo 3 são apresentadas algumas propostas de melhoramento e optimização do mesmo. O ambiente de experimentação é apresentado no capítulo 4, bem como uma breve descrição da implementação dos protocolos em análise. O capítulo 5 contém os resultados dos testes realizados bem como a análise dos mesmos, e o capítulo 6 discute uma possível implementação em ambiente real da solução Trusted Peers. Um capítulo de resumo, conclusões e apresentação de pistas para o trabalho futuro fecha este documento.

2. O protocolo BitTorrent

O protocolo denominado BitTorrent foi criado em 2001 pelo americano Bram Cohen com o objectivo de disseminar grandes quantidades de dados, organizados sobre a forma de ficheiros, de forma colaborativa, e o seu funcionamento é descrito em [4]. O protocolo tornou-se extremamente popular porque distribui a carga da transferência por todos os participantes em vez de sobrecarregar e congestionar servidores cujo custo de operação é muitas vezes elevado. Outros aspectos, que o tornaram popular face a alternativas anteriores, são o facto de combater os *free-riders* e de rentabilizar a capacidade de *upload*[1] de todos os participantes na disseminação de um ficheiro. Com efeito, os dados a transferir são divididos em blocos que são trocados entre os participantes, de uma forma que combate o uso abusivo do sistema para único proveito individual.

O protocolo é relativamente complexo e passa por vários algoritmos e estratégias diferentes que variam a cada momento da transferência. O seu funcionamento e as razões do seu sucesso são abordados nas próximas secções.

2.1 Publicar conteúdo

Antes de se poder proceder à partilha de um ficheiro por vários participantes é necessário criar um ficheiro contendo meta-dados do mesmo. Este ficheiro de meta-dados tem o nome de *torrent* e agrega informação como o nome do ficheiro a partilhar, o seu tamanho, o número de blocos em que se divide, um código de *hash* dos diferentes blocos, que permite verificar a sua integridade, e o URL (Uniform Resource Locator) do *tracker*, um servidor cujo funcionamento será abordado na secção 2.2.

O ficheiro, com a extensão `.torrent` e de tamanho reduzido, deve então ser publicado de forma a que todos os interessados lhe tenham acesso. Geralmente isso é feito colocando-o num repositório público (*online*), especializado no alojamento de ficheiros *torrent*, dos quais `www.mininova.org`, `http://thepiratebay.org` ou `www.btjunkie.org` são bons exemplos, embora a disseminação do ficheiro possa ser feito por qualquer meio, desde que o mesmo se torne acessível aos interessados.

[1] *Upload* designa o tráfego de dados no sentido ascendente, isto é, do computador local para o computador remoto, e pode ser traduzido por carregar. Para facilitar a leitura utiliza-se o termo inglês.

2.2 O Tracker

Apesar do protocolo BitTorrent funcionar com base numa rede totalmente distribuída, para um participante entrar nessa mesma rede precisa de um ponto de acesso conhecido. Esse ponto de acesso é fornecido por um servidor dedicado que mantém informação referente aos utilizadores que estão a descarregar um determinado ficheiro. Este servidor tem o nome de *tracker* e deve estar acessível sempre que novos participantes se pretendem juntar à rede do ficheiro que este serve.

Quando um participante pretende juntar-se a uma rede de partilha, contacta o *tracker*, comunicando através de um protocolo muito simples que assenta sobre HTTP ("Hypertext Transfer Protocol"), enviando informação referente ao ficheiro em que está interessado, assim como informações adicionais acerca da sua ligação à Internet. O *tracker* responde então com uma lista contendo cerca de 50 contactos aleatórios de outros participantes activos e interessados em partilhar o mesmo ficheiro, e com os quais o participante pode estabelecer conexões para assim se inserir na rede e iniciar ou recomeçar a transferência.

O *tracker* não interfere posteriormente na transferência dos dados, apenas mantém informação referente aos participantes e outras estatísticas.

2.3 A partilha de ficheiros

Depois de criado um *torrent* do ficheiro em causa e de um *tracker* estar em funcionamento para aquele ficheiro, é necessário que exista pelo menos uma cópia integral do ficheiro entre todos os participantes na partilha do mesmo. Qualquer participante com uma cópia completa do ficheiro na rede é designado de *seed*, em português semente. Todos os outros participantes que não possuem a totalidade do ficheiro são denominados de *leechers*, termo não traduzido para português, pelo que estes participantes são simplesmente denominados de utilizadores. Os participantes estão ligados uns aos outros de forma aleatória e comunicam constantemente aos seus vizinhos os blocos que possuem. Desta forma, os utilizadores sabem a quem podem pedir os blocos que desejam. Existem vários mecanismos para determinar quais os blocos a pedir, a que participantes e com que frequência. Estes mecanismos serão abordados nas secções 2.4 e 2.5.

Para acelerar o processo de transferência dos blocos, geralmente com a dimensão de 256kB,

estes são ainda divididos em pedaços mais pequenos, sub-blocos de 16kB, e vários pedidos desses mesmos sub-blocos são feitos em paralelo aos diferentes vizinhos. Os tamanhos dos blocos e sub-blocos podem variar conforme a implementação do protocolo. Desta forma consegue-se maximizar a largura de banda disponível. Quando são transferidos todos os blocos que compõem o ficheiro o utilizador torna-se semente, e por omissão fica a responder a pedidos de blocos, até se desligar da rede.

A transferência do ficheiro é assim feita recorrendo a diversos participantes para obter todos os blocos necessários. É esta estratégia que permite aliviar a carga da semente inicial e garantir transferências mais rápidas quando comparadas com transferências de uma fonte singular. No entanto, o comportamento altruísta de oferecer blocos não surge de forma voluntária entre os participantes na partilha, este é imposto por mecanismos abordados na secção 2.5.

2.4 Selecção de blocos

A ordem pela qual são pedidos os blocos é de grande importância no desempenho do sistema. Em primeiro lugar é dada prioridade aos sub-blocos de um bloco, isto é, apenas são feitos pedidos a sub-blocos de outro bloco depois de terem sido pedidos todos os sub-blocos do primeiro bloco. Desta forma completam-se blocos o mais depressa possível. O algoritmo para determinar a ordem pela qual devem ser pedidos os blocos é denominado de *Rarest First*[2], e consiste em ser dada prioridade aos blocos mais raros entre os parceiros. Esta estratégia muito simples tem várias implicações bastante favoráveis:

- Se um utilizador detém os blocos mais raros a probabilidade de ser possível fazer *upload* para qualquer outro utilizador é grande. Esta propriedade é importante para manter activos utilizadores com boa largura de banda.

- Ao deixar para o fim os blocos mais comuns aumenta-se a probabilidade de concluir com sucesso a transferência, sendo mais fácil obter os blocos em falta.

- Ao obter primeiro os blocos mais raros o sistema torna-se independente da semente original mais depressa, garantindo a existência de pelo menos uma cópia de cada bloco entre todos os participantes de forma bastante rápida.

[2] *Rarest First* pode ser traduzido por "o mais raro em primeiro lugar".

Em determinados momentos o algoritmo de selecção de blocos não é exactamente o descrito por razões de optimização. Esses momentos são abordados nos próximos pontos.

2.4.1 Momento inicial

Quando se inicia a transferência, utilizar o algoritmo de *Rarest First* não seria óptimo pois os blocos mais raros são geralmente detidos por um menor número de participantes, tornando assim a sua transferência mais morosa. Como tal, inicialmente são feitos pedidos de blocos de forma aleatória até que o participante disponha de 4 blocos completos.

Desta forma já é possível usar estes primeiros blocos como moeda de troca, ou seja, disponibilizá-los a outros utilizadores e assim conseguir obter blocos deles. Após concluída a transferência dos primeiros 4 blocos aleatórios a política muda para *Rarest First*.

2.4.2 Momento final

Quando são chegados os instantes finais da transferência e todos os sub-blocos em falta já foram pedidos, a política de *Rarest First* deixa de ser válida. Para evitar desacelerar a transferência devido a um possível participante mais lento, quando já foram pedidos todos os sub--blocos em falta são repetidos esses mesmos pedidos por todos os participantes conhecidos, minimizando assim a possibilidade de uma transferência mais lenta. Esta estratégia é denominada por "modo de fim de jogo" (*end game mode*). À medida que os sub-blocos vão sendo recebidos os pedidos para os mesmos sub-blocos vão sendo cancelados para poupar alguma largura de banda. Há no entanto sempre algum desperdício, mas este não tem muito peso face à totalidade do ficheiro pois esta fase dá-se durante um curto espaço de tempo.

2.5 O algoritmo de Choking

Não basta a um participante estabelecer ligações para garantir que vai receber pacotes de outros participantes. Cada participante é responsável por maximizar a sua própria taxa de *download*[3]. Isto é conseguido fazendo *download* de todos os participantes que conseguirem e decidindo para que participantes fazer *upload*. Para cooperar um participante faz *upload*, para

[3] *Download* designa o tráfego de dados no sentido descendente, isto é, do computador remoto para o computador local e pode ser traduzido por descarregar.

não cooperar um participante faz *choke*[4]. *Choke* significa uma recusa em efectuar *upload*, não invalida qualquer *download* desse mesmo participante nem fecha a conexão estabelecida.

É através do algoritmo de *Choking* que se implementa uma política de *tit-for-tat*[5], garantindo uma troca justa entre todos os participantes na partilha do ficheiro. Só assim se garante o bom funcionamento do sistema. O objectivo final do algoritmo de *Choking* é alcançar um equilíbrio no qual não seja possível efectuar uma alteração e todos os participantes ficarem em melhor posição. Um exemplo: se dois participantes com boa largura de banda têm fracas taxas de *download* podem simplesmente parar de fazer *upload* para outros participantes e dirigi-lo um para o outro, conseguindo subir o rendimento do *download* de ambos. É nessa base que o algoritmo de *Choking* actua. Do ponto de vista de um único participante podemos então descrever o algoritmo como dirigir o *upload* para os participantes que têm melhores taxas de *upload* para com o próprio. Obtêm-se então várias conexões bidireccionais, sendo estas as melhores encontradas até ao momento. Por omissão são apenas 4 mas este valor depende muito da implementação e pode ser alterado até mesmo dinamicamente no decorrer da transferência.

Surge então a questão de como decidir quais os melhores parceiros. A taxa medida é apenas a de *download*, e apenas é considerada a média dos últimos 10 segundos. São necessários alguns segundos para poder medir com relativa certeza a taxa de *download* de uma conexão, pelo que são utilizadas rondas de 10 segundos, ao fim dos quais se avaliam as conexões e efectuam possíveis trocas, mantendo as alterações até ao inicio da ronda seguinte. 10 segundos por ronda é tempo suficiente para uma conexão TCP estabilizar e atingir a velocidade máxima, originando medições adequadas.

Um participante escolhe assim sempre os melhores parceiros com base no *upload* que estes lhe dirigem, e responde-lhes com a melhor taxa de *upload* que puder oferecer. No entanto, é possível que existam outros parceiros capazes de lhe oferecer uma taxa de *upload* melhor que os actuais. Surge então uma estratégia de exploração de parceiros através de *optimistic unchokes*, ou *unchokes* optimistas, em que um parceiro escolhido aleatoriamente é *unchoked*, independentemente da sua taxa de *upload*. Este processo executa em rondas de 30 segundos, tempo suficiente para que o parceiro seleccionado responda o melhor que possa à taxa de *download* oferecida. Na prática demonstra-se que esta estratégia funciona, e obtém um bom rendimento na procura de melhores parceiros.

Este algoritmo de *Choking* faz com que os participantes tenham tendência a manter ligações

[4] *Choke* pode ser traduzido por estrangular.
[5] *Tit-for-tat* é uma expressão inglesa que representa uma troca directa.

com participantes com largura de banda igual ou superior à deles próprios. Se estiverem ligados a um participante com largura de banda inferior têm tendência a encontrar alguém melhor. Se estiverem ligados a um participante com largura de banda superior, este reciprocamente tem tendência a encontrar um participante melhor e desistir daquela ligação. A figura 2.1 ilustra esta propriedade. Inicialmente temos 4 utilizadores, onde dois têm uma taxa de *upload* de 2 MBPS (a verde) e os outros dois 4 MBPS (a azul), ilustração da esquerda. Nesta situação os utilizadores com 4 MBPS conseguem uma taxa de *download* de 2 MBPS. Se após *optimistic unchoke* um utilizador com 4 MBPS de taxa de *upload* descobre o outro, estes terão preferência em efectuar *upload* um para o outro pois obtêm assim uma taxa de *download* de 4 MBPS cada um, o que resulta na ilustração da direita.

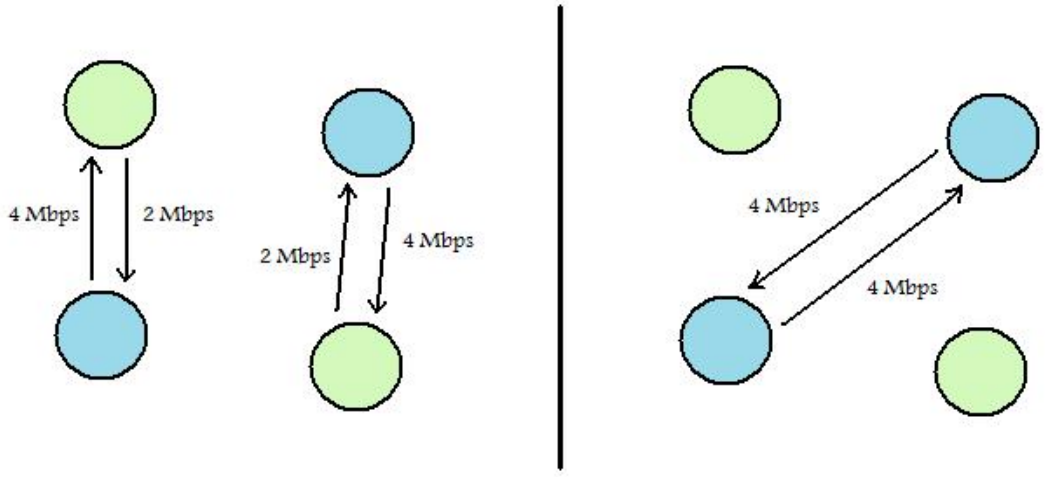

Figura 2.1 Exemplo ilustrativo da tendência em manter ligações com utilizadores com largura de banda igual ou superior, no algoritmo de *Choking*. À esquerda a situação inicial, à direita a situação após *optimistic unchoke*.

Ocasionalmente um participante pode ser *choked* por todos os participantes com quem estava activamente a transferir blocos. Este estado traduz-se em péssimas taxas de *download* até que novos participantes sejam descobertos por *unchoke* optimista. De forma a corrigir esta situação, quando um participante não recebe qualquer sub-bloco de outro participante durante um minuto assume que foi *choked* e corta por completo o *upload* para esse participante, excepto em situação de *unchoke* optimista. Desta forma liberta largura de banda que pode aplicar noutros participantes e descobrir parceiros mais interessantes mais rapidamente.

Uma excepção no algoritmo de *choke* é quando um participante se encontra em estado de semente. Não havendo *download* para medir tem de ser aplicada outra regra para decidir quais

os participantes a quem enviar blocos. Em versões anteriores do algoritmo essa escolha era feita com base no *upload*. Eram escolhidos os participantes que maximizavam a taxa de *upload* do emissor, ou seja, os utilizadores com maior largura de banda. Este algoritmo era facilmente alvo de exploração pois um utilizador com boa largura de banda podia monopolizar as sementes, rapidamente completar a transferência do ficheiro e sair logo de seguida, sem oferecer qualquer bloco a outro participante. Para colmatar esta falha grave, o algoritmo para quando um participante se encontra no estado de semente foi redesenhado e funciona do seguinte modo: a cada ronda de 10 segundos, a lista de participantes é reordenada, ficando em primeiro lugar os participantes que foram *unchoked* há mais tempo. São então escolhidos os 3 primeiros participantes e 1 participante aleatório dessa lista para receberem blocos. Excepcionalmente de 30 em 30 segundos não é escolhido um participante aleatório, sendo escolhidos os primeiros 4 da lista. Este algoritmo funciona muito melhor que anterior pois garante a rotatividade do acesso às sementes, servindo melhor todos os participantes e corrigindo os problemas que levaram à exploração do protocolo em versões anteriores. Esta correcção ao algoritmo apenas está presente na versão 4.0.0 e posteriores.

2.6 Razões para o sucesso

Existem inúmeras aplicações que implementam o protocolo do BitTorrent, e a enorme quantidade de utilizadores que as usa é um testemunho do sucesso que este alcançou. De seguida indicam-se algumas razões que levaram ao sucesso do BitTorrent.

2.6.1 Facilidade de utilização

Para o utilizador comum utilizar o BitTorrent para descarregar ficheiros da Internet é simples e fácil. Ao contrário de muitos predecessores o BitTorrent conta com pesquisas fáceis e rápidas, baseadas em plataformas já existentes e com as quais o utilizador já está familiarizado, como uma convencional página Web.

Embora actualmente a interface do BitTorrent já esteja relativamente mais complexa do que inicialmente, trata-se de uma aplicação extremamente simples de usar, e não requer grandes conhecimentos informáticos por parte do utilizador. Todo o funcionamento do protocolo está escondido do utilizador, que apenas vê a transferência a decorrer por meio da percentagem

completa, velocidade de *download* e velocidade de *upload*. Não requer qualquer manutenção dos ficheiros a transferir ou gestão de ligações a *trackers* ou parceiros, simplificando para o utilizador a utilização do BitTorrent.

2.6.2 Robustez

O BitTorrent tem demonstrado ter bastante robustez perante situações que se revelaram fatais para muitos sistemas que o antecederam.

O protocolo escala muito bem, e é comum existirem redes de partilha que ultrapassam as dezenas de milhar de utilizadores, em ficheiros de dezenas de GB, sem se registar qualquer problema entre os utilizadores. Sobrevive mesmo às denominadas *flash crowds*, que consistem numa grande afluência de utilizadores num curto espaço de tempo e demonstra que a politica *tit-for-tat* funciona bem.

O protocolo também resiste a abusos, penalizando os *free-riders* com baixas taxas de *download*, e mesmo as soluções que procuram explorar algumas falhas no protocolo não o conseguem fazer sem contribuir para a comunidade, como o exemplo explorado em [10], em que a aplicação BitTyrant tenta optimizar o *download* gerindo de forma dinâmica o número de participantes com os quais mantém ligações, mas para que a ligação seja bidireccional precisa de oferecer uma taxa de *upload* mínima e responder aos pedidos desses participantes.

Apesar de no geral ser robusto, é necessário referir que o protocolo tem um ponto de estrangulamento no *tracker*, o ponto agregador de todo o sistema. Todos os participantes que desejam começar ou recomeçar a transferência têm de entrar em contacto com o *tracker*, e este tem de estar sempre disponível. Assim, além de sofrer com *flash crowds*, o *tracker* está ainda sujeito a ataques de *denial of service*[6], que tiram partido deste ser o ponto de estrangulamento e atacam todo o sistema. Ainda assim estes problemas não são significativos pois cada *torrent* pode ter um *tracker* diferente, e um dos clientes mais utilizados, o Azureus-Vuze [17], usa uma DHT (Distribuited Hash Table) para distribuir as funcionalidades do *tracker*, minimizando assim a escala destas falhas.

[6] *Denial of service* designa um tipo de ataque conhecido por incapacitar o atacado de responder aos pedido que lhes venham a ser dirigidos. Normalmente é feito bombardeando o atacado com pedidos falsos, deixando os reais não atendidos.

2.6.3 Comunidades

Foram criadas inúmeras comunidades de partilha de ficheiros, grande parte delas especializadas (filmes, software, séries televisivas, etc.), e nas quais os participantes são incentivados a permanecer ligados como sementes após concluírem a transferência. Isto gera comunidades com grandes números de ficheiros disponíveis para partilha e tempos de transferência reduzidos, levando assim à entrada de mais membros e difundindo o sucesso do protocolo.

Existem também comunidades fechadas que requerem que os utilizadores se registem. Esta perda de privacidade traz como benefício uma forma de combate suplementar aos *free-riders*. Este pode ser feito através de duas técnicas. Uma primeira recorre a um sistema de pontuação associado a cada conta de utilizador. Para poder descarregar conteúdo é necessário trocar pontos, pontos esses que se ganham mantendo-se no estado de semente. Outra técnica, mais usada devido à sua simplicidade, é manter registo da quantidade de bytes que cada utilizador faz *upload* e *download*. Para simplificar fundem-se os dois valores numa medida conhecida como rácio, definida por *upload*/*download*. O rácio toma valores inferiores a 1 caso o utilizador faça mais *download* que *upload* e maior que 1 caso contrário. Para ambas as técnicas existem restrições para os utilizadores que colaboram menos e recompensas para os que colaboram mais.

As restrições são feitas ao nível dos conteúdos a transferir, no caso da pontuação se um utilizador não tem pontos suficientes para efectuar a transferência, esta é-lhe impossibilitada. No caso do rácio, com um rácio inferior a um determinado valor todas as transferências são bloqueadas até que o utilizador regularize o seu rácio por meio de *upload*. Normalmente estas regras apenas se aplicam após o utilizador ter um determinado valor de pontos ou conteúdo descarregado, para assim dar oportunidade a novos utilizadores de entrarem na comunidade. Utilizadores mal intencionados poderiam explorar este aspecto e criar novas contas sempre que as anteriores sejam bloqueadas. Para prevenir essa exploração, a maioria das comunidades tem regras rígidas para a entrada de novos utilizadores, como longos períodos de espera, registos apenas por convite, etc.

As recompensas podem ser variadas, desde acesso a conteúdo exclusivo, condições especiais nas transferências ou mesmo recompensas apenas na comunidade como títulos de reconhecimento e opções extra. Isto vem encorajar os utilizadores a ligarem-se apenas no estado de semente, para assim aumentar a sua pontuação ou rácio.

A instituição de todos estes mecanismos faz-se com recurso à implementação de um cliente de código fechado e *trackers* privados, aos quais apenas o cliente tem acesso. É o *tracker* que

gere as estatísticas dos utilizadores como a pontuação e o rácio através de mensagens trocadas com o cliente em intervalos regulares. O cliente requer a autenticação do utilizador antes de iniciar comunicações com o *tracker*, e após obter as informações necessárias pode impor restrições e atribuir recompensas sempre que seja o caso.

Nazareno Andrade *et al.* mostram em [1] como estes mecanismos de restrições e recompensas são capazes de gerar comunidades muito activas, com tempos de transferência reduzidos e torrents capazes de existir durante muito tempo, beneficiando toda a comunidade.

2.7 Análises

Vários artigos apresentam análises do comportamento e do desempenho do protocolo em diversos ambientes. Estes são referidos em seguida.

O estudo [7] demonstra a ocorrência de vários fenómenos durante a vida de um *torrent*. Inicialmente dá-se um efeito de *flash crowd*, com muitos participantes a juntarem-se à rede. Este período inicial é reconhecido por um número baixo, mas crescente de sementes, e um maior número de utilizadores com um crescimento superior. Algum tempo depois o número de utilizadores atinge o seu máximo e começa a decrescer enquanto o número de sementes continua a aumentar, alcançando o seu máximo algum tempo depois. Após este período inicial ambos os valores decrescem para níveis muito mais baixos que têm tendência a manter-se constantes durante o resto da vida do *torrent*. O mesmo estudo demonstra também a importância das sementes, que assumem a maior parte da carga da transferência do ficheiro, especialmente depois do momento inicial de *flash crowd*.

Um outro estudo, [12], abordou a robustez do sistema e detectou vários pontos fracos, sendo o mais preocupante destes a nível dos *trackers*, que podem ser alvo de ataques e falhas e tornam todo o sistema mais vulnerável. Existem estratégias para minimizar esses problemas, mas o estudo também evidenciou um outro problema do sistema: o facto do suporte físico (espaço para alojar os servidores contendo os *trackers*) ser em muitos casos mantido através de doações, o que pode por si só ser um risco de segurança. O mesmo estudo aborda também a opinião dos utilizadores, e é notório como mesmo num período caracterizado por diversas falhas, não só do *tracker* mas também do site onde os torrents eram publicados, não foi registada qualquer quebra no número de utilizadores após as falhas, provando assim a popularidade do sistema. É também abordada a problemática da poluição dos ficheiros, algo que foi terrível para o sistema Kazaa,

e que o sistema BitTorrent consegue, até ao momento, fazer face sem problemas de maior. A solução é simples: verificação manual por parte dos utilizadores da veracidade e utilidade dos ficheiros que estão a ser partilhados. Parece uma tarefa complicada de incentivar, mas a verdade é que na maioria das comunidades esta tarefa é desempenhada por voluntários, que, através de comentários que partilham com a comunidade, conseguem ter um óptimo rendimento na tarefa de identificar ficheiros falsos ou maliciosos. Concretamente, no estudo foram feitas várias tentativas de poluir o sistema com ficheiros falsos e em todas elas os utilizadores com a tarefa de monitorizar o conteúdo dos ficheiros detectaram e baniram o conteúdo em causa, demonstrando uma eficácia de 100%.

2.8 Em síntese

O protocolo BitTorrent separa-se do conceito da procura do conteúdo e foca-se na transferência do mesmo, tentando que esta seja feita com o auxílio de toda uma comunidade, trabalhando em conjunto para o mesmo fim: a partilha entre todos de determinado conteúdo. Desta forma a procura de ficheiros é externa ao sistema, e apenas existe um ponto agregador simples, uma forma de entrar na partilha, constituída por um simples servidor para cada ficheiro a partilhar.

O funcionamento do protocolo é bastante complexo, recorrendo a diferentes algoritmos para optimizar os diversos momentos de vida da transferência, e tentando adaptar-se o melhor possível às condições existentes, de recursos de rede e de participantes. Uma das maiores contribuições para o seu funcionamento é um mecanismo instaurado que garante que todos os utilizadores contribuem para o sistema, sob pena de transferências lentas e que não aproveitam os recursos de rede.

O protocolo alcançou já muito sucesso, contando com inúmeras implementações e um crescente número de utilizadores, mas não cai vítima do seu sucesso. Robusto e capaz de lidar com redes de partilha enormes prova como o seu sistema de *tit-for-tat* funciona, embora haja espaço para melhoramentos, como os abordados no próximo capítulo.

3 . Optimização e evoluções do algoritmo original

Desde que foi criado em 2001 vários aspectos do protocolo foram alvo de estudos, melhoramentos e adaptações. São abordados nas secções que se seguem alguns desses estudos e melhoramentos.

3.1 BitTyrant

Michael Piatek *et al.* observaram em [10] que o algoritmo que regula a troca de blocos pode ser alvo de exploração, e para o comprovarem desenharam um sistema capaz de aproveitar esses pontos fracos para proveito próprio.

O sistema é implementado tendo em conta três estratégias interligadas entre si. A primeira é de que se pretende estar ligado aos participantes com melhor largura de banda. Isso é determinado através de medições da taxa concedida pelos participantes, ou no caso de não ter recebido ainda nenhum bloco do mesmo, estimada através da taxa a que divulgam possuir novos blocos. A segunda estratégia passa por oferecer o mínimo indispensável a cada participante para manter a reciprocidade. Através da largura de banda estimada do parceiro é possível calcular a probabilidade de este se manter recíproco caso se diminua a taxa de *upload*. Isto leva-nos à terceira estratégia, diminuindo a taxa de *upload* é possível alocá-la em novas conexões, maximizando assim a quantidade de participantes de quem se recebe blocos. Para poder aplicar estas estratégias o número de ligações simultâneas tem de ser dinâmico.

Com base numa implementação popular e de código aberto do protocolo BitTorrent, Azureus[17], os autores procederam às alterações necessárias para criar o seu cliente e testaram-no em ambientes reais e no Planet-Lab [11], sendo também testado no último qual o resultado se todos os participantes utilizassem o cliente modificado.

Os resultados obtidos em ambientes reais foram bastante reveladores, com um ganho médio de desempenho na ordem de 1,72, e onde 25% das transferências concluíram 2 vezes mais rapidamente quando comparadas com o cliente sem alterações.

Os resultados obtidos no Planet-Lab, em que todos os participantes usam o cliente modificado foram também dignos de análise, registando-se duas situações distintas: quando existe uma boa percentagem de utilizadores com larguras de banda elevadas o desempenho global do sistema aumentou quando comparado com o cliente sem alterações, mas estes valores não

se verificaram quando se limitaram as larguras de banda dos clientes, tendo-se mesmo registado um degradar do desempenho. Este fenómeno é explicado pelo melhor uso da largura de banda do cliente modificado em utilizadores com uma largura de banda elevada. Quando esta é limitada todo o sistema sofre, e nesses casos o cliente sem alterações consegue um melhor desempenho, pois oferece mais largura de banda aos participantes com quem mantém ligações.

3.2 BitMax

Nikolaos Laoutaris *et al.* tomam em [8] uma abordagem baseada em redes cooperativas, em que o objectivo é disseminar os dados o mais rápido possível, como em ambientes empresariais quando é necessário actualizar software. Assim sendo, em vez do algoritmo de *choke*, os autores propõem que os participantes maximizem a sua taxa de *upload* pelos participantes a quem se ligam, transmitindo mais blocos completos para outros participantes, acelerando assim a transferência a um nível global. Trata-se de um algoritmo totalmente altruísta e facilmente explorável por utilizadores mal-intencionados, mas que traz vantagens ao nível da velocidade global de transferência.

A solução apresentada, denominada de BitMax, foi construída através de uma implementação do BitTorrent, em que o algoritmo de *choking* foi substituído pelo seguinte: em vez de dividir a largura de banda por um determinado número de ligações, procurar saturar a ligação de um participante, e repetir para o máximo de participantes possíveis enquanto o emissor tiver largura de banda disponível. É dada preferência aos participantes com maior largura de banda e menos blocos pois estes podem acelerar ainda mais a transferência a nível global.

Tendo em conta uma possível distribuição do BitMax ao público, foram sugeridas alterações para lidar com o problema do *free-riding*, nomeadamente a distribuição sobre a forma de cliente fechado. Desta forma os utilizadores não tinham acesso às configurações do programa de modo a impedi-lo de enviar informações correctas como velocidade de *download* e *upload*. *Free-riding* neste sistema teria então de usar programas auxiliares que serviriam de filtros entre o tráfego trocado entre os participantes, mas mesmo essa solução poderia ser combatida através da crifra das mensagens trocadas, acrescentando assim a robustez necessária para assegurar o sucesso do BitMax.

O sistema foi testado num simulador de sistemas *peer-to-peer*, GPS [18], e foram registados tempos de transferência 2 vezes mais rápidos quando comparados com o actual sistema

BitTorrent. Foi ainda notório que a disseminação de blocos é mais rápida, tornando o sistema independente da semente inicial mais rapidamente.

3.3 Biased Neighbor Selection

Ruchir Bindal *et al.* em [3] analisam o tráfego gerado pelo sistema BitTorrent, nas suas diversas aplicações, que atravessa as ligações entre os ISPs (Internet Service Provider) e observam que este é significativo. Estudando o sistema BitTorrent verificam ser possível diminuir este tráfego entre ISPs de forma significativa sem diminuir o desempenho do mesmo. Abordando o problema pelo lado do ISP propõem soluções com o objectivo de minimizar o tráfego entre ISPs. Controlando as ligações entre parceiros, limitando-as essencialmente a participantes dentro do mesmo ISP conseguem reduzir a quantidade de dados que tem de atravessar as ligações entre ISPs. Devem no entanto existir sempre ligações para participantes fora do mesmo ISP para garantir o acesso a todos os blocos dos dados a transferir.

Apresentam assim duas propostas de implementação distintas: uma primeira que consiste em alterar a implementação do algoritmo e do *tracker*. Modificando o *tracker* para que devolva preferencialmente participantes do mesmo ISP e um número menor de participantes fora do ISP. Caso não existam participantes no mesmo ISP em número suficiente o *tracker* informa o cliente que o deve tornar a contactar dentro de determinado período temporal para tentar obter novos participantes do mesmo ISP. Determinar qual o ISP em que cada participante se insere pode ser feito por um dos seguintes métodos:

- O *tracker* ter acesso a mapas de topologia de internet para determinar o ISP através dos endereços IP.
- Os ISPs publicarem aos *trackers* quais os seus intervalos de endereços IP.
- Como o protocolo BitTorrent trabalha sobre HTTP (Hyper Text Trasfer Protocol) a *proxy* HTTP do ISP poderia acrescentar um novo header à mensagem definindo qual o ISP de origem.

A segunda proposta de implementação passa por dispositivos de *traffic shaping*[1] inteligentes. Neste caso a estratégia a seguir seria manter uma listagem de todos os participantes do ISP a

[1]Dispositivos de *traffic shaping* actuam sobre os pacotes que atravessam o canal e tomam acções no sentido de alterar a prioridade dos mesmos, ou eventualmente suprimi-los.

participar na rede de partilha e, sempre que um novo utilizador dentro do ISP deseja entrar na rede de partilha interceptar a mensagem do *tracker* e substituir participantes externos por participantes internos ao ISP. Este sistema pode ainda actuar sobre participantes com listas de parceiros mais desactualizadas, terminando as suas conexões, forçando assim o participante a pedir uma nova lista de parceiros ao *tracker*, a qual pode então manipular de forma a incorporar os novos parceiros.

Através de simulação os autores observaram o cumprimento dos objectivos a que se propuseram: a redução do tráfego entre ISPs. Apenas foi observada uma situação adversa, explicada pelo algoritmo de *Rarest-first*: quando a rede formada dentro dos ISPs é pequena, todos os participantes têm uma visão global da rede interna ao ISP e têm assim tendência a pedir blocos a participantes de fora do ISP, que oferecem blocos mais raros na visão singular de cada participante.

Registou-se uma melhoria de desempenho nos tempos de transferência quando comparados com o BitTorrent em situações de *traffic-shaping* por parte dos ISPs, minimizando ao mesmo tempo o tráfego que atravessa o ISP. No caso de não existir *traffic-shaping* pelos ISPs o desempenho mantém-se equivalente ao original.

3.4 TRIBLER

J. A. Pouwelse *et al.* abordam em [13] os sistemas de partilha de ficheiros *peer-to-peer* segundo um novo paradigma, o das redes sociais. Em vez de se focarem nos aspectos técnicos da partilha de ficheiros focam-se nos aspectos sociais, nos utilizadores, para fornecer um serviço mais apelativo e ao mesmo tempo mais eficaz. Apoiados no sistema BitTorrent tentam ainda introduzir inovações ao nível dos pontos centrais de falha (*tracker* e pesquisa de ficheiros) e ainda na transferência em si através de uma técnica que denominam de *download* colaborativo.

O sistema final acrescenta assim vários módulos ao BitTorrent como bases de dados de conhecidos, preferências e localização para oferecer uma experiencia nova ao utilizador, mantendo o sistema compatível com as redes tradicionais de partilha que usam implementações do BitTorrent.

O sistema é então implementado com base numa implementação corrente do protocolo BitTorrent, à qual são acrescentados vários módulos que originam os 5 componentes base:

- ***Map***. Através de um motor de pesquisa geográfica é capaz de fornecer a localização dos

parceiros.

- ***Downloads***. Usando o protocolo BitTorrent ou através de *Download* Colaborativo, este componente gere as transferências.

- ***My Friends***. Componente que gere os amigos através de listas de amigos e *Social Networking*. Este componente também tem um papel activo nos *Downloads* Colaborativos.

- ***Files I Like***. Este componente permite ao utilizador avaliar todos os ficheiros descarregados com uma nota de 0 a 5, e através desta nota oferecer ao utilizador conteúdo relacionado com os seus gostos e preferências.

- ***Taste Buddies***. Este último componente, intimamente ligado com o anterior, permite dar a conhecer utilizadores com gostos semelhantes a ele próprio. Com base nas avaliações feitas no componente *Files I Like* são escolhidos utilizadores que podem ser adicionados à lista de amigos.

A descoberta de parceiros para a transferência pode ser feita através do método original do protocolo BitTorrent (contactando o *tracker*) ou através do módulo *Buddycast*, que usando algoritmos epidémicos troca preferências e contactos entre parceiros e assim consegue encontrar parceiros interessados em transferir os mesmos ficheiros. A pesquisa de ficheiros a transferir pode igualmente ser feita através deste sistema. Sem acrescentar um *overhead* significativo quando comparado com o sistema BitTorrent, o modulo *Buddycast* acrescenta descentralização ao sistema, evitando os *bottlenecks* e riscos de segurança presentes nos *trackers* e a necessidade de soluções externas presentes na actual forma de procura de ficheiros a transferir (recorrendo a sites web agregadores de *torrents*).

Com uma rede social montada é possível usar os amigos para acelerar a transferência de ficheiros usando-os como ajudantes, aquilo que é designado de *Download* Colaborativo. Os ajudantes ligam-se na mesma rede de partilha, mesmo que não estejam interessados no ficheiro, e vão transferindo blocos do ficheiro que redireccionam para o interessado. No contexto das experiencias descritas em [13], usando o *Download* Colaborativo foi possível tornar uma transferência 2 a 3 vezes mais rápida quando comparada com a transferência normal. Este comportamento totalmente altruísta por parte dos ajudantes é conseguido somente nas redes sociais através de relações pessoais, que podem ou não basear-se no facto de que quando precisarem os ajudantes serão ajudados pelos amigos. Este algoritmo é abordado com maior detalhe no estudo de Garbacki *et al.* [6] e descrito na secção seguinte.

Apesar de oferecer um sistema de incentivos baseado em redes sociais, este sistema é facilmente alvo de *free-riding* por parte de utilizadores mal intencionados. É relativamente fácil obter transferências rápidas pedindo ajuda a utilizadores sem devolver essa mesma ajuda mais tarde. Caso comecem a existir penalizações devido ao comportamento incorrecto basta criar uma nova identidade para continuar o comportamento abusivo.

3.5 2Fast

Pawel Garbacki *et al.* demonstram em [6] e [5] que é possível melhorar o tempo de transferência através de um protocolo novo por eles desenvolvido e que assenta sobre o BitTorrent. Propõem uma troca de largura de banda em vez de trocar blocos dos ficheiros a transferir. Baseando-se em redes sociais desenharam um sistema que permite obter ajuda de outros participantes na transferência do ficheiro. Quando um utilizador deseja transferir determinado conteúdo pode pedir ajuda a utilizadores amigos, também chamados de ajudantes, que cedem a sua largura de banda inutilizada para transferir blocos para o primeiro, também designado por colector. O colector coordena um qualquer número de ajudantes para que possam transferir blocos utilizando o protocolo BitTorrent original acelerando a transferência quando comparada com o BitTorrent sem alterações.

Não existe uma troca imediata de largura de banda, os ajudantes fazem-no com base na confiança de que mais tarde os papéis se vão inverter e poderá contar com a ajuda do colector para acelerar as suas transferências. Este modelo apenas é viável em sistemas como o Tribler [13], com autenticação, redes sociais e avaliação entre utilizadores. No entanto os autores estudaram uma possível aplicação do sistema fora do Tribler.

O sistema foi implementado alterando uma implementação popular do algoritmo do BitTorrent, o BitTornado, e acrescentando os mecanismos necessários para que este novo protocolo funcione.

O funcionamento é relativamente simples. O colector inicia uma transferência normal de BitTorrent, e ao mesmo tempo contacta um certo número de parceiros informando qual o ficheiro em questão. Esses parceiros, caso aceitem ajudar na transferência, tornam-se ajudantes e iniciam então a mesma transferência, também de forma normal, mesmo que não estejam interessados no conteúdo. Antes de iniciarem a transferência de um bloco verificam se o colector

não o possui e se não está já nenhuma outro ajudante a transferi-lo. Quando um ajudante termina a transferência de um bloco transfere uma cópia para o colector. Podem-se considerar estes ajudantes como *proxies* do colector, multiplicando-o assim na rede de partilha e dividindo o tempo de transferência por todos.

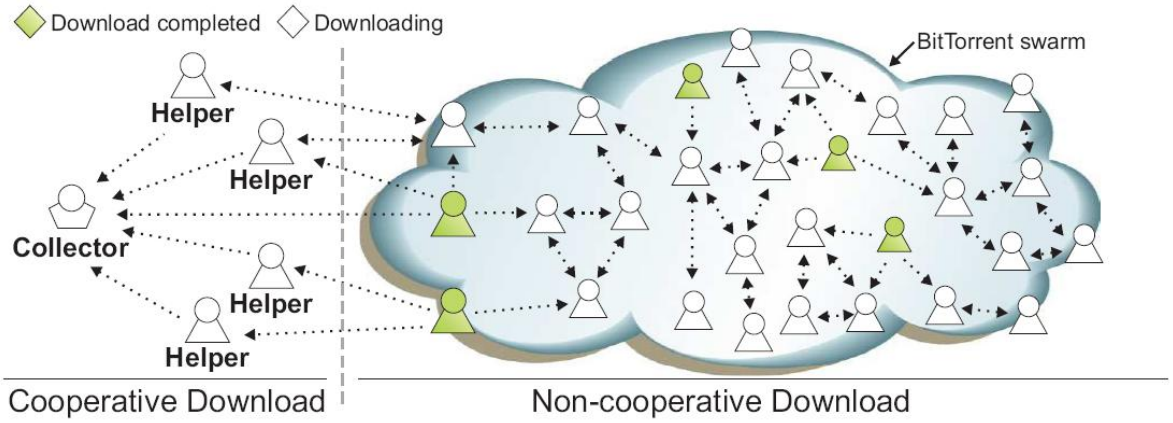

Figura 3.1 Representação de uma rede baseada no protocolo 2Fast. Retirado de [13].

Todos os blocos transferidos dos ajudantes para o colector são contabilizados e entram na dívida do colector para cada um dos ajudantes, que será paga mais tarde da mesma forma, servindo de ajudante para qualquer outro parceiro.

Além do funcionamento descrito na secção anterior foi implementado um mecanismo para melhorar o desempenho dos ajudantes. Uma vez que os momentos iniciais e finais do protocolo BitTorrent são sempre mais lentos devido à falta de blocos para troca e escassez dos blocos em falta respectivamente, de modo a minimizar estes problemas foram criadas duas extensões ao protocolo 2Fast que permite melhorar esses pontos.

Uma primeira extensão permite que os ajudantes transfiram blocos que já tenham sido transferidos por outros ajudantes para os ajudar nas trocas. A carga de transferir estes blocos extra não é comunicada ao colector, pelo que estas transferências são da responsabilidade dos ajudantes, e como tal são estes que definem qual o seu grau de altruísmo em relação a este ponto.

Uma segunda extensão permite aos ajudantes trocar mensagens entre eles com as suas listas de parceiros e quais os blocos que estes possuem de modo a melhorar a troca de blocos. Esta extensão vem possibilitar acelerar bastante a transferência como demonstrado nos resultados.

Segundo as experiencias descritas em [6], o sistema foi testado em simulação e em ambiente real, sendo os resultados obtidos bastante próximos, o que leva a concluir que a simulação foi

bem realizada. A versão base do protocolo 2Fast demonstrou melhorias significativas face aos tempos de transferência conseguidos apenas com o sistema BitTorrent. Com as extensões que levam à troca de parceiros e blocos entre os ajudantes, os autores mediram tempos até 3.5 vezes melhor que o protocolo BitTorrent, uma melhoria bastante significativa e muito próximo do tempo óptimo calculado teoricamente.

3.6 PACE: Peer-Assisted Content Exchange

Christina Aperjis *et al.* analisam em [2] o protocolo BitTorrent segundo uma abordagem económica, especialmente no que diz respeito à troca de blocos, que segue um espírito de troca-directa, e detectam existirem diversas situações de falha do modelo de mercado subjacente. Seguindo o modelo económico é possível melhorar estas situações com uma troca bilateral baseada num modelo regulado pelo mercado. Assim propõem-se melhorar o desempenho do BitTorrent com recurso a noções de economia.

Atribuindo ao *upload* o papel de moeda de troca, é possível quebrar algumas regras impostas pelo modelo actual, as ligações passam a ser multilaterais, é assim possível fazer *upload* agora para o participante X e mais tarde *download* do participante Y com o saldo obtido do *upload* para X. É ainda tida em consideração a topologia da rede atribuindo custos diferentes às diversas ligações, que são mais caras conforme o número de nós de rede que têm de atravessar, gerando assim uma selecção de parceiros mais ponderada, beneficiando os ISPs (Internet Service Provider).

O sistema, denominado de PACE ("Price-Assisted Content Exchange") é implementado recorrendo a 4 componentes distintos:

- Um cliente para vender ficheiros, no qual os utilizadores podem divulgar os ficheiros que pretendem partilhar, e onde o seu preço é calculado.

- Um cliente para comprar ficheiros, onde os utilizadores podem ver que ficheiros estão a ser partilhados e qual o seu preço. Se tiverem saldo suficiente podem iniciar a transferência dos ficheiros seleccionados.

- Um serviço de preços de rede gere os custos de rede que devem ser adicionados ao cálculo dos preços de cada utilizador.

- Um servidor central que mantém registo do saldo de cada utilizador e garanta o pagamento entre todos os utilizadores. Esta componente actuando como banco e bolsa é crucial em ambientes em que os utilizadores desconfiem uns dos outros.

Com todos estes componentes é possível proceder à partilha de ficheiros conforme desenhado pelos autores. Os preços são calculados dinamicamente e variam conforme a procura. Se um ficheiro é muito procurado e há pouca oferta, o preço sobe, se existe muita oferta e pouca procura, o preço desce. Aos preços calculados é somado o custo de rede, dado pela componente correspondente. Este aspecto pode ser afinado de modo a tornar a rede mais eficiente do ponto de vista de transferências entre ISPs diferentes.

Os utilizadores deste sistema têm de se autenticar perante a entidade gestora dos saldos para garantir que o sistema pode funcionar correctamente. Novos utilizadores, com o saldo a zero, têm a oportunidade de efectuar transferências a custo zero, mas apenas de certos conteúdos e durante um curto espaço de tempo. Podem assim obter ficheiros que poderão usar para aumentar o seu saldo para poderem efectuar as transferências que assim desejarem. Este período inicial não é sustentável, e pode potenciar ao *free-riding*, mas como se trata de um período muito curto, que existe apenas uma única vez na vida de cada utilizador, não tem muito impacto no sistema.

O sistema é assim capaz de oferecer incentivos aos utilizadores para permanecerem ligados em modo de semente, recompensando-os com saldo que poderão aplicar mais tarde de forma a acelerar as suas transferências. Desta forma é possível aumentar o desempenho do sistema, tornando-o mais aliciante a novos utilizadores.

Através das simulações descritas em [2] os autores observaram que o sistema combate eficazmente os *free-riders*, demonstrando que os não *free-riders* concluem a transferência 60% mais rapidamente. Observaram ainda que o sistema mantém a mesma robustez perante *flash-crowds* que o algoritmo original do BitTorrent, e que o mecanismo de custos de rede actua bem, reduzindo o número de ligações entre ISPs beneficiando participantes localizados no mesmo ISP.

3.7 PropShare

David Levin *et al.* tomam uma abordagem alternativa às trocas feitas entre participantes na rede de partilha em [9], onde, após análise ao *tit-for-tat*, o decidem trocar por um sistema de leilões baseado em rondas, tornando as trocas mais justas e resistentes a ataques por parte

de clientes estratégicos. Adicionalmente analisam também o processo de revelação de blocos, e propõem um comportamento segundo uma estratégia que visa incrementar o interesse dos outros participantes nos seus blocos. Chamaram a este novo cliente PropShare.

O sistema baseia-se então na seguinte visão: um participante vai leiloar a sua largura de banda durante um curto período de tempo. As ofertas são as contribuições de outros parceiros nos últimos 4 períodos de tempo. Assim os vencedores serão aqueles que ofereceram mais ao participante nos últimos períodos e adicionalmente um participante aleatório, como forma de descobrir novos parceiros. Desta forma é dada preferência aos participantes que estão a oferecer blocos ao participante, incentivando assim ao *upload* entre todos os participantes, prejudicando os *free-riders* que apenas recebem blocos durante os curtos períodos em que são descobertos.

Este novo algoritmo traz bastantes vantagens mas também tem algumas falhas. Rondas demasiado longas podem prejudicar o desempenho dos participantes. Nota-se também que a convergência dos participantes para uma situação estável pode ser por vezes demorada. Apesar disto, a generalidade dos testes efectuados pelos autores mostram o novo cliente como mais rápido que o BitTorrent normal, e até mesmo que o BitTyrant [10].

A outra estratégia proposta pelos autores foi a escolha selectiva de que blocos anunciar como disponíveis. Partindo do princípio que um participante pretende manter o maior número de participantes interessados em si de forma a obter mais *upload*, mostrar todos os blocos que possuí pode não ser a melhor estratégia. A figura 3.2 ilustra os diferentes cenários possíveis na óptica de interesse para o participante i. As setas representam interesse entre os participantes. Estão ordenados de forma decrescente de interesse para i, onde o cenário (a) representa o cenário com maior interesse e (f) o cenário com menor interesse. A ideia torna-se mais clara com o exemplo em que o participante i possui 2 blocos raros no sistema, e partilha um deles com j e o outro com k. Na ronda seguinte j poderia trocar o primeiro com k e k o segundo com j, tornando assim o participante i desinteressante para ambos (cenário (f) da figura 3.2). Nesta situação a melhor estratégia seria esconder o segundo bloco, e apenas o anunciar quando necessário de forma a manter o interesse (mantendo assim o cenário (a) da figura 3.2 por mais tempo). Isso é conseguido através de um algoritmo de análise dos blocos que os parceiros possuem e oferecendo apenas os necessários para manter o interesse, calculando quais os mais raros entre os vizinhos e guardando esses para anunciar mais tarde. Esta estratégia mantém o interesse no participante durante mais tempo e consegue taxas de *download* superiores quando comparadas com o protocolo original.

Os autores testaram o seu protocolo, bem como a estratégia da escolha selectiva de blocos

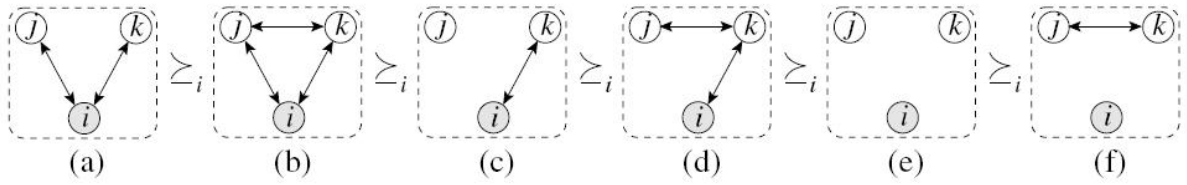

Figura 3.2 Cenários possíveis na óptica de interesse para o participante *i*. Estão ordenados de forma decrescente de interesse para *i*. As setas representam interesse. Retirado de [9].

a anunciar e, segundo os cenários descritos em [9] registaram uma ligeira melhoria de tempos dos clientes PropShare face aos clientes com o BitTorrent original. Registaram também que a existência dos primeiros pode beneficiar o desempenho de todo o sistema, sendo assim viável o seu lançamento para o público.

Este novo sistema combate assim de forma mais eficaz o *free-riding* quando comparado com o BitTorrent, mas não elimina o problema, uma vez que ainda pode ser alvo do mesmo através das descobertas optimística de parceiros. É no geral mais justo nas trocas, mas não oferece mecanismos de incentivos à permanência dos utilizadores na rede.

3.8 FairTorrent

Alex Sherman *et al.* apresentam em [16] uma alteração simples ao algoritmo do BitTorrent, mas capaz de garantir trocas mais justas, combater o *free-riding* e ao mesmo tempo alcançar grandes melhorias de desempenho.

A alteração é efectuada no algoritmo de *chocking*. É abandonado o conceito de manter 4 (ou qualquer outro número) ligações para manter em estado *unchoked* baseadas na taxa de *upload*. Em vez disso todas as ligações são mantidas em estado *unchoke* e os seus pedidos recebidos. Os pedidos vão sendo atendidos por uma ordem baseada no cálculo dos *bytes* em dívida. Isto é, sempre que o participante recebe *bytes* do participante X incrementa um contador de *bytes* em dívida. O participante guarda um destes contadores para cada participante conhecido. A lista é reordenada, por ordem crescente de dívidas, a cada bloco enviado, e seleccionado qual o participante a quem enviar o próximo bloco. Caso a capacidade de *download* do primeiro participante na lista seja esgotada, ou este não esteja interessado em nenhum dos blocos possuídos, avança-se para o seguinte na lista. São enviados em paralelo os blocos necessários para saturar

a capacidade de *upload* do participante, maximizando assim a utilização de recursos.

Neste novo protocolo não existe a noção de ronda, e como tal não existe uma descoberta optimista de novos parceiros como no BitTorrent. A descoberta de novos participantes é feita de forma automática ao percorrer a lista. Caso não exista retorno suficiente da parte dos primeiros participantes da lista ordenada os restantes vão ganhando prioridade e os seus pedidos são atendidos. Num relativo curto espaço de tempo o participante irá obter uma taxa de *download* igual à sua taxa de *upload*, e é isto que torna as trocas efectuadas pelo FairTorrent mais justas.

Esta alteração permite uma aproximação entre participantes com larguras de banda semelhantes muito rápida, formando assim ligações entre participantes de forma equilibrada e aumentando o desempenho dos mesmos. Este objectivo já tinha sido tentado por outras aproximações [9], mas esta proposta alcança resultados ainda melhores quando comparados com os anteriores.

O mecanismo de descoberta de novos parceiros do FairTorrent tem implicâncias muito positivas no *free-riding*. Já não são oferecidos muitos *bytes* através de *optimist unchokes*, na melhor das hipóteses um *free-rider* apenas obterá poucos *bytes* de cada um dos seus vizinhos que não são sementes, e apenas uma única vez. Os únicos participantes a oferecer blocos sem olhar aos contadores de dívidas são as sementes, que oferecem blocos de uma forma alternada entre todos os participantes que conhecem. É importante manter as sementes com este comportamento para que novos utilizadores possam transferir os seus primeiros blocos e enviá-los para outros participantes de forma a obter reciprocidade.

Uma outra vantagem do FairTorrent está na sua simplicidade. Não conta com grandes alterações ao algoritmo base, mantendo a retro-compatibilidade com o algoritmo original do BitTorrent, não realiza estimativas de largura de banda, cálculos complexos ou alterações a parâmetros em tempo de execução. As alterações, segundo os autores, não passam de cerca de uma centena de linhas no código original do BitTorrent, prova mais que evidente desta simplicidade.

Os autores testaram a sua proposta contra diferentes versões do algoritmo BitTorrent nas mais diversas condições conforme descrito em [16], e concluem que o FairTorrent permite tempos de transferência até 5 vezes melhores e ganhos de desempenho do sistema entre 60% a 100%. Concluem ainda que devido a estes mesmos ganhos de desempenho começa a tornar-se possível a utilização de sistemas P2P (nomeadamente o FairTorrent) para *streaming* de vídeos, embora seja ainda uma área a necessitar de alguma investigação.

3.9 Em síntese

As propostas aqui abordadas foram desenvolvidas com o propósito de tornar o BitTorrent mais justo e/ou melhorar o desempenho global do mesmo. Estes objectivos foram alcançados através de alterações sobretudo no algoritmo que gere as trocas entre participantes (casos do BitTyrant, BitMax, PACE, PropShare e FairTorrent). Comprova-se assim que o algoritmo de *Choking* é o principal ponto de intervenção de modo a criar alternativas capazes de combater mais eficazmente o *free-riding* e ao mesmo tempo melhorar o rendimento global de uma rede de partilha.

Poucas destas propostas abordam no entanto a temática dos incentivos. O número de participantes presentes numa rede de partilha, especialmente a participação altruísta, é importante para o rendimento global do sistema. Como tal, incentivar a permanência de utilizadores é uma área que merece a devida investigação. Duas das propostas estudadas, o 2Fast e o PACE, são capazes de oferecer incentivos aos utilizadores. A primeira comporta incentivos sociais, dificilmente mensuráveis, enquanto a segunda implementa incentivos que estão directamente relacionados com as transferências, e como tal é possível verificar o seu impacto em testes. Ambos estes sistemas partilham no entanto uma desvantagem: o aumento da complexidade de implementação do algoritmo.

Verifica-se com agrado existirem esforços nesta área para combater o *free-riding* e melhorar o rendimento global desta família de sistemas P2P. É importante eliminar o *free-riding* dos sistemas P2P pois este consome recursos sem devolver nada à rede, diminuindo assim o desempenho do sistema e comprometendo os tempos de todos os outros utilizadores, o que pode levar a uma quebra de popularidade como já se verificou noutros sistemas. Um sistema P2P sem *free-riding* é um sistema mais rápido, mais justo e consequentemente mais atractivo a novos utilizadores.

Existe ainda um vasto caminho a explorar no que toca às optimizações do algoritmo BitTorrent. Seja o número de ligações que se mantém com os parceiros, a largura de banda disponibilizada a cada um deles, ou mesmo a sua escolha, há sempre espaço para melhoramentos como provam os estudos aqui abordados. Independentemente do caminho escolhido é satisfatório ver que é possível obter bons resultados quer no combate ao *free-riding*, quer na melhoria de desempenho do sistema.

4 . O ambiente de experimentação

4.1 Simuladores, emuladores e a Internet

Qualquer trabalho de investigação requer validação. Essa validação é alcançada por meio de experiências que nos confirmam, ou não, o alcançar dos objectivos propostos. Testar um sistema distribuído e complexo como um sistema *peer-to-peer* não é fácil e requer o auxílio de ferramentas capazes de lidar com a enorme quantidade de variáveis e informação gerada. Numa perspectiva singular há já muito a considerar. Individualmente cada participante tem de gerir ligações, escolher os melhores parceiros e lidar com um grande número de blocos de um ficheiro de dimensão geralmente considerável. Se olharmos para o sistema de uma forma global, a quantidade de informação dispara. Além de todos os nós a considerar, multiplicando assim as tarefas anteriores, é necessário parametrizar e monitorizar as ligações usadas por estes. Tudo isto forma um sistema difícil de gerir, mais difícil quanto maior a dimensão usada.

A Internet é uma infra-estrutura extremamente vasta, complexa e variada. Efectuar portanto testes de sistemas que executem sobre a Internet torna-se um desafio acrescido. Existem, no entanto, várias ferramentas que possibilitam testar os sistemas antes de estes serem lançados no ambiente real de execução. Podemos separá-los em dois grandes grupos: simuladores e emuladores.

Os simuladores são sistemas de *software* fáceis de gerir como um todo. Permitem utilizar um modelo de uma rede, que se pode parametrizar, de modo a aproximá-la do modelo pretendido, seja uma pequena rede empresarial ou a Internet global. Tudo é virtual, como tal requer que os algoritmos e o sistema a testar sejam introduzidos no simulador, o que geralmente requer codificá-lo de acordo com a filosofia do mesmo. Normalmente correm localmente numa mesma máquina, requerendo máquinas capazes de lidar grandes quantidades de dados num curto espaço de tempo.

Os emuladores são sistemas que imitam o comportamento de uma rede com recurso a virtualização de nós e da rede, externa à solução em teste, isto é, montando uma infra-estrutura de emulação que irá regular as ligações de diversas máquinas virtuais através de uma rede emulada. O ponto forte dos emuladores é que não requerem alterações da aplicação ou solução a testar, pois as máquinas virtuais podem executar directamente o código da aplicação em teste. Um bom exemplo de emulador é o ModelNet [15]. O ModelNet emula uma rede alterando as ligações entre as máquinas virtuais de modo simular que estão ligadas através da mesma.

Existe ainda uma outra ferramenta, mais próxima do ambiente real que simuladores e emuladores, o Planet-Lab [11]. Trata-se de uma vasta rede de máquinas espalhadas pelas mais diversas instituições de investigação do mundo como universidades, institutos, etc., ligadas entre si através da Internet, formando desta forma uma rede real de tamanho considerável, onde o comportamento das aplicações pode ser testado.

O teste das propostas a desenvolver neste trabalho é de importância vital, pois apenas assim será possível validar as soluções propostas e verificar qual o seu impacto efectivo no desempenho global do sistema.

Inicialmente foi abordada a hipótese de realizar os testes experimentais num ambiente de emulação proporcionado pelo ModelNet [15]. Seria necessário preparar uma máquina com o software do ModelNet, a qual seria parametrizada de acordo com determinado modelo de rede, e à qual seriam ligadas N máquinas, cada uma correndo várias máquinas virtuais com implementações do BitTorrent. Inicialmente este cenário constituía uma alternativa interessante pois permitiria obter resultados mais próximos da realidade que com um simulador e proporcionava um contacto directo com uma implementação real do algoritmo BitTorrent, contudo acarretava alguns pontos desfavoráveis:

- A instalação e configuração do sistema não são triviais. No entanto este ponto pode-se considerar ultrapassado pois já é possível contar com uma instalação de ModelNet no laboratório de Redes do Departamento, pronta a ser utilizada.

- Seria necessário dotar toda e cada máquina (virtual ou não) presente na emulação de uma implementação e configuração do BitTorrent.

- Finalmente levanta-se a questão da monitorização dos recursos e recolha de dados, que iria requerer técnicas adicionais, incrementando ainda mais a complexidade do sistema e o tempo necessário para a realização dos testes.

Assim, depois de uma análise das alternativas disponíveis, atendendo ao objectivo de encontrar uma ferramenta relativamente simples de usar e que oferecesse facilidades na recolha de dados, foi escolhida a solução de usar a simulação.

4.2 O simulador escolhido

Para o trabalho foi escolhido o simulador GPS-P2P (General Purpose Simulator for P2P Network) [18] [19], criado em 2005 por Weishuai Yang and Nael Abu-Ghazaleh, totalmente implementado na linguagem Java, com o objectivo de auxiliar a comunidade científica a testar a emergente onda de sistemas *peer-to-peer*, em especial o BitTorrent, alvo de uma grande quantidade de estudos e adaptações.

O que levantou de imediato a atenção para este simulador foi o facto de, ao contrário de muitos outros simuladores que, para simularem o envio de uma mensagem de A para B se limitam a entregar a mensagem a B no tempo estipulado pelos parâmetros de rede, este simular o protocolo TCP (Transmission Control Protocol) de forma mais realista, dividindo os canais por todas as ligações que os atravessam e ajustando as respectivas larguras de banda. Este aspecto é de grande importância para os sistemas *peer-to-peer* do tipo do BitTorrent, daí a necessidade da criação de um novo simulador.

Os autores além de criarem o simulador implementaram no mesmo o protocolo BitTorrent, facilitando assim muito a utilização do *software* por terceiros, uma vez que a documentação do código é escassa, e poupando muito tempo de implementação.

Aliando a estes pontos fortes o facto de o simulador ser utilizado por um estudo como o apresentado em [8], já abordado no ponto 3.2, tornou a escolha do GPS como ferramenta de testes para este trabalho a mais adequada, apesar dos problemas surgidos a seguir, como veremos na próxima secção.

4.3 Melhoramentos introduzidos no simulador

Na sua versão original, apesar das vantagens sobre outros sistemas, o simulador apresentava alguns problemas e não oferecia todas as funcionalidades necessárias aos cenários que pretendíamos testar. Foi então necessário introduzir alguns melhoramentos para adaptá-lo às necessidades do trabalho. Estes melhoramentos e alterações são apresentados e justificados de seguida:

- **Redes heterogéneas:** Originalmente o simulador parametrizava toda a rede com os mesmos valores de largura de banda (excepto os nós de trânsito). Como se pretendeu poder estudar redes com uma distribuição heterogénea das larguras de banda, foram criadas

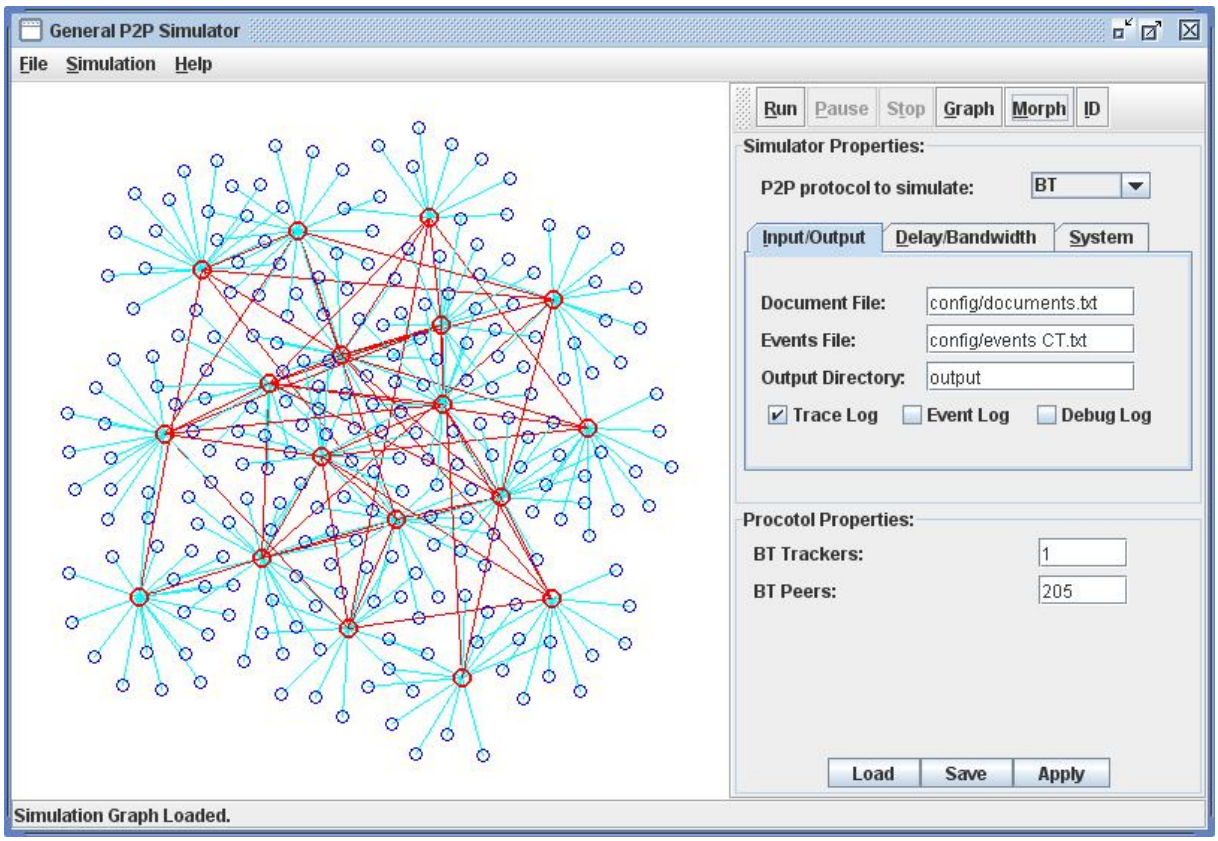

Figura 4.1 Interface do simulador GPS

funções para alterar as larguras de banda de alguns nós da rede segundo alguns parâmetros a fornecer pelo utilizador. Essas funções actuam aquando da criação da rede física e alteram os valores de alguns canais segundo uma probabilidade definida nos referidos parâmetros, criando assim redes com uma percentagem de nós com uma largura de banda diferente.

- **Simulações de grandes dimensões:** Nos testes iniciais não era possível realizar simulações com redes de muitos nós, nem com ficheiros muito grandes pois esgotavam-se rapidamente os limites físicos do hardware. Após investigação observou-se que era possível realizar uma optimização numa estrutura sem perder as suas funcionalidades. Esta estrutura encontrava-se no objecto do tipo documento e guardava informação sobre quais os participantes que já detinham determinado ficheiro, sendo usada apenas para o cálculo dos blocos mais raros. À medida que os participantes completavam a sua transferência esta estrutura guardava mais e mais referências para participantes, crescendo exponencialmente. A solução foi substituir a lista de nós detentores de cada bloco por um inteiro, incrementado sempre que um participante anunciava a recepção de um bloco. Esta alteração mantém as funcionalidades iniciais, continuando a potenciar um cálculo correcto dos blocos mais raros. Não havia necessidade de guardar nesta estrutura informação referente a que participantes detêm os blocos pois esta estrutura apenas é usada para determinar quais os blocos mais raros, os pedidos são efectuados sobre aqueles que os participantes com quem se mantêm ligações oferecem. Esta nova estrutura levanta um novo risco, o de contabilizar um participante mais que uma vez por cada bloco, no entanto esse problema não se coloca pois neste ambiente controlado sabe-se que nenhum participante envia mensagens falsas e não há perda de mensagens.

- **Comportamento após a transferência:** Na versão original do simulador todos os participantes permaneciam ligados como sementes após terminarem a transferência até ao final da simulação ou até receberem eventos de controlo para abandonarem a sessão. O anterior mecanismo não se adequava ao pretendido pois os referidos eventos de controlo eram executados num momento definido antes da simulação, e dado a imprevisibilidade do tempo de conclusão de uma transferência era impossível ter participantes a abandonar a rede assim que concluída a transferência. Uma vez que se pretendiam efectuar testes em ambientes de escassez de recursos foi adicionada uma verificação para quando terminada a transferência permanecer em estado de semente ou terminar de imediato a sessão.

- **Estatísticas:** Apesar de já vir presente no código alguma recolha e tratamento de estatísticas, estas revelaram-se insuficientes para o trabalho e como tal foram acrescentados métodos para recolher vários outros dados importantes nos mais diversos locais. São recolhidos dados relativos aos blocos trocados entre os diversos intervenientes na rede de partilha, com divisão explícita dos diferentes papéis tomados (sementes, utilizadores e outros abordados no âmbito de protocolos muito específicos). Há ainda um tratamento dos resultados finais, capaz de registar os tempos de transferência médios, mais rápidos e mais lentos, mais uma vez, para cada classe de participantes na rede de partilha. A contagem de blocos é feita através de contadores específicos sempre que um bloco é enviado ou recebido. O tratamento dos tempos de transferência é realizado por uma classe específica que actua sobre o ficheiro contendo o *trace* resultante da simulação.

- **Entidade central:** Um dos pressupostos deste trabalho é a existência de uma entidade central segura e conhecida por todos os participantes, capaz de ser contactada para recolher estatísticas e oferecer uma serie de informações, conforme requisitado por cada protocolo. Esta entidade foi criada e é instanciada no arranque de cada simulação.

- **Gestão de novos parâmetros:** Algumas destas alterações introduziram novos parâmetros alteráveis. Estes estão presentes numa única classe como variáveis estáticas de forma a ser possível alterá-los facilmente e sem necessidade de percorrer o código. Não estão ainda presentes na interface gráfica, ficando esse passo para melhoramentos futuros.

- **Redes assimétricas:** O simulador original apenas permite efectuar testes com linhas simétricas, o que distancia de alguma forma os resultados obtidos dos reais em que a taxa de *upload* e *download* têm sempre diferenças consideráveis, e limitam mesmo a velocidade do sistema. Isto deve-se ao facto de existir uma falha na definição a nível lógico das conexões. Quando **A** estabelece uma conexão para **B** esta é feita sobre determinados canais. Se **B** precisar de estabelecer uma ligação para **A** ao invés de abrir uma nova conexão reaproveita a já existente, sem validar os canais que esta atravessa. A correcção deste problema envolve alterar a definição de conexão, bem como todos os métodos para a sua criação, registo e acesso. Esta alteração apesar de importante não foi efectuada devido à complexidade das intervenções a efectuar.

Estas alterações foram introduzidas à medida que se revelaram necessárias e representam um esforço significativo pois requereram o estudo detalhado de partes significativas do código

do simulador, tal como será a seguir descrito brevemente.

4.4 Funcionamento do simulador

O simulador segue o paradigma de programação por eventos, em que cada mensagem é representada como um evento a ser processado e encaminhado por um *scheduler*, que mantém uma lista ordenada temporalmente com todos os eventos a processar no futuro. Eventos e mensagens confundem-se no simulador como um único conceito. É possível existirem eventos entre os mais diversos objectos, incluindo os próprios, basta para tal possuir funções para efectuar o tratamento dos mesmos. Alguns dos eventos podem ser definidos num ficheiro externo à simulação, que é depois usado para gerar os eventos no tempo de execução virtual definido no ficheiro. Neste ficheiro encontram-se os eventos para iniciar e parar a transferência, bem como outras eventuais acções específicas a outros protocolos, que envolvem tomadas de decisões externas à execução do protocolo. A execução dos protocolos no simulador lança os restantes eventos, como as mensagens trocadas entre os diversos nós da rede.

O simulador apresenta uma interface gráfica simples e intuitiva, onde são apresentados diversos parâmetros da simulação que podem ser alterados, bem como uma representação visual da rede e dos seus nós, com colorações diferentes para nós de trânsito, nós inactivos e nós participantes na rede de partilha. Os parâmetros adicionais criados durante a execução deste trabalho e descritos brevemente na secção anterior não se encontram visíveis na interface gráfica. Observam-se também na interface botões que permitem iniciar e pausar ou parar a simulação. Quando uma simulação está em curso é ainda possível observar as taxas de utilização de cada ligação em tempo real e a percentagem de conclusão da transferência em cada nó, tal como ilustrado na figura 4.2

A modelação da rede pelo simulador funciona com recurso a objectos que gerem, em cada ligação, a largura de banda e as conexões que a atravessam. Sempre que a nível lógico há um evento de início ou final da transferência de um bloco, estes objectos recalculam a largura de banda disponível, simulando desta forma o mecanismo de partilha do canal existente no protocolo TCP-IP. Esta modelação não é no entanto isenta de erros e arredondamentos, sendo os valores obtidos apenas aproximações. Este mecanismo apenas é aplicado à transferência de blocos, as mensagens de controlo do protocolo são trocadas ignorando a capacidade de rede que consomem.

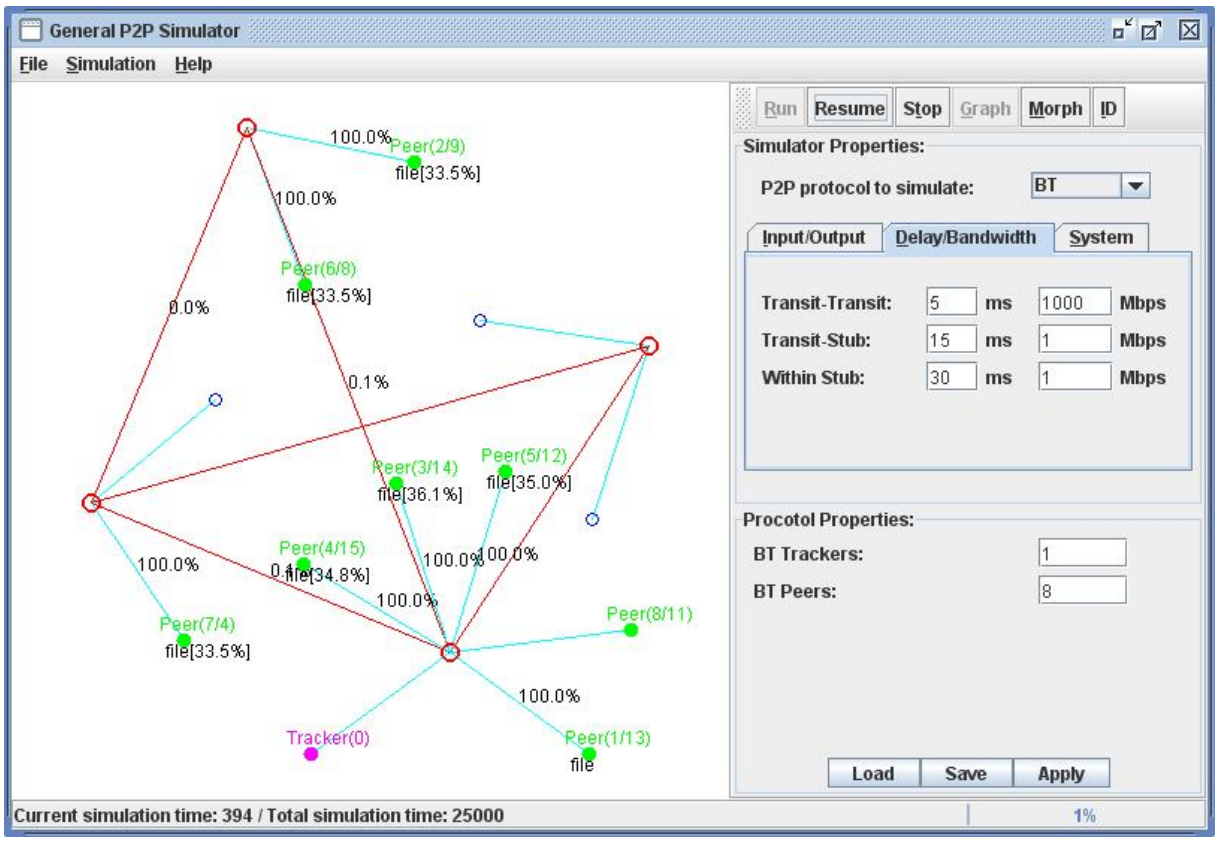

Figura 4.2 Interface observável durante uma simulação no simulador GPS

O simulador trazia já uma implementação funcional do protocolo BitTorrent. Esta é bastante complexa, contando com dezenas de classes, algumas das quais serão descritas de seguida, dado a sua importância na implementação e funcionamento do algoritmo.

A principal entidade responsável por uma transferência é a classe BTSession. Cada nó tem uma instância da classe BTSession por cada transferência em curso. A BTSession é responsável por coordenar e agregar toda a informação referente à transferência para a qual foi criada. Mantém objectos onde guarda referencias para as ligações estabelecidas, quais os participantes a quem está ligada, qual o estado da transferência, o ficheiro a ser transferido, a sua visão local da rede, entre outras. É ainda a classe BTSession que trata todas as mensagens referentes ao protocolo BitTorrent (com excepção da de início da transferência, que é tratada pelo nó), agindo em conformidade, ou encaminhando-as para os objectos que necessitam de as tratar. Possui a maioria dos métodos necessários à execução do algoritmo e funciona como classe agregadora e coordenadora da execução do mesmo.

As ligações entre participantes são geridas por instâncias da classe BTSocket. Os objectos BTSockets mantêm referência para os participantes entre os quais está estabelecida a conexão, qual a largura de banda disponível e, caso exista uma transferência de bloco em curso, informação do bloco que está a ser transferido. Para o objecto do tipo BTSocket são encaminhadas as mensagens de recepção de bloco, ou de início de transferência de bloco, para que os ajustes à largura de banda possam ser efectuados, e os métodos de tratamento do bloco sejam chamados no respectivo objecto.

O ficheiro que está a ser transferido é gerido por uma instância da classe BTDocument, que mantém para cada bloco a percentagem já transferida. Possui os métodos que dão continuidade à recepção de um bloco (iniciados numa instância da BTSocket) e inicia os processos que devem ser desencadeados sempre que um bloco é transferido, como por exemplo anunciar que já se possui aquele bloco ou dar por concluída a transferência.

Os algoritmos de *Choking* e selecção de blocos são mantidos em classes fora da BTSession, que são acedidas por instâncias da mesma sempre que há necessidade.

A troca de blocos é totalmente baseada em eventos despoletados através da troca de mensagens. A transferência de um bloco é iniciada por uma de duas formas: ou o participante recebeu uma mensagem de *unchoke* de outro participante e vai então enviar um pedido, ou terminou de receber um bloco e vai pedir outro ao mesmo participante até receber uma mensagem de *choke* ou encontrar um participante melhor.

A recepção de um bloco é também efectuada através da troca de mensagens. Ao receber

uma mensagem do tipo bloco, a BTSession calcula o tempo necessário para receber o bloco, atendendo à largura de banda disponível na conexão através da qual o irá receber, recorrendo à instância BTSocket associada para obter os dados necessários. Agenda então para o tempo de chegada uma mensagem a ser tratada pela instância de BTSocket, que por sua vez tratará de executar os métodos necessários ao tratamento do bloco na instância de BTDocument associada à referente instância de BTSession. A mensagem de recepção de bloco na BTSocket pode ser adiada se as condições de largura de banda se alterarem.

Esta é uma descrição simplificada do funcionamento do simulador, bem como do protocolo BitTorrent já presente no mesmo. É importante ter presente o funcionamento do simulador e a sua implementação do BitTorrent para compreender a implementação de algumas alterações e outros protocolos, que foram implementados e testados, e que serão abordados na próxima secção.

4.5 Soluções e alterações ao BitTorrent que foram implementadas

Nesta secção é abordada a implementação de soluções e alterações ao algoritmo do BitTorrent no simulador escolhido para este trabalho. De forma a facilitar a leitura está dividido em subsecções, uma por cada solução ou alteração.

4.5.1 Protocolo 2Fast

O protocolo 2Fast, tal como descrito em [6] por Pawel Garbacki et al., para além das melhorias de desempenho demonstradas pelos autores, pode oferecer a base para um sistema baseado em incentivos. Surgiu assim o objectivo de o implementar no simulador para verificar quais os ganhos de desempenho na óptica dos diferentes intervenientes neste protocolo.

A descrição do protocolo em maior pormenor pode ser encontrada na secção 3.5 ou em [6], e de forma muito resumida consiste na criação de 2 novos papéis na rede de partilha: Collector e Helper, não alterando em nada o protocolo BitTorrent, sobre o qual assenta. O papel de Helper pode ser tomado por qualquer utilizador com largura de banda disponível, aplicando-a para ajudar a transferência de um Collector, mesmo que não esteja interessado no conteúdo que o Collector está a obter. O papel de Collector é tomado por utilizadores que desejam transferir determinado conteúdo de forma rápida, contando para tal com a ajuda de um

determinado número de Helpers, que actuarão como suas *proxies* na rede de partilha, sem pedir nada em troca.

É precisamente nesta diferença de papéis que pode assentar uma política de incentivos, recompensando de forma positiva os utilizadores que tomem o papel de Helpers, e requerendo alguma forma de pagamento da parte dos utilizadores que tomam o papel de Collectors. Outra alternativa seria oferecer a possibilidade de se tornarem Collectors apenas aos utilizadores comprovadamente merecedores, como meio de incentivar um comportamento altruísta. A discussão destas opções é independente da implementação do sistema.

Para implementar o protocolo 2Fast foi necessário estender as classes BTSession, BTDocument e BTAlgorithmPieceSelection para cada um dos novos papéis, e aí proceder a alterações profundas.

No caso dos Helpers foi necessário dotar a classe Peer de novas funcionalidades, nomeadamente a de se voluntariar para ajudar e esperar ser contactado por um Collector, só então iniciando a sessão. O início da sessão começa com a configuração dos canais directos para o Collector, instâncias da classe BTSocket. As alterações começam ao nível da classe BTAlgorithmPieceSelection, onde têm de ser tomados em consideração os blocos em que o Collector está interessado. A classe BTDocument estendida teve assim de ser dotada de novas estruturas capazes de distinguir entre o que efectivamente o Helper possuí e aquilo que apenas o Collector possuí. Por último a classe BTSession estendida teve de ser dotada de estruturas e métodos capazes de lidar com as novas estruturas e métodos das extensões às classes já mencionadas, bem como do tratamento de todas as novas mensagens trocadas neste novo protocolo e verificações adicionais para evitar usar o Collector como um participante normal ao efectuar trocas.

Para os Collectors as alterações focaram-se nos mesmos mecanismos e classes, mas com uma complexidade maior na classe BTSession, onde foi necessário coordenar não só a sua própria sessão, com tudo o que isso implica, mas também todas as sessões dos Helpers associados. Foi necessário criar métodos para verificar o número de Helpers e se necessário procurar mais. A classe BTSession estendida foi dotada de estruturas capazes de gerir filas de espera por cada Helper, contendo os blocos que cada um está pronto a enviar. A classe BTAlgorithmPieceSelection estendida teve de passar a contar com os blocos que os Helpers já transferiram na selecção de novos blocos, e a classe BTDocument estendida teve de passar a distinguir os blocos transferidos, os transferidos pelos Helpers e os não transferidos.

Para não aumentar a já crescente complexidade, o protocolo foi implementado da forma mais simples que possível. Após estabelecidas as conexões entre Helpers e Collectors, cada um

deles executa o protocolo original do BitTorrent com as diferenças de que os Helpers não estão interessados em transferir blocos em que o Collector não está interessado, e os Collectors por sua vez não estão interessados em transferir blocos que já são possuídos pelos Helpers. Sempre que o Collector inicia a transferência de um bloco anuncia-o aos Helpers, que deixarão de estar interessados nesse bloco. Quando um Helper termina a transferência de um bloco anuncia--o ao Collector, que, se estiver interessado nesse bloco, anunciará aos Helpers não estar mais interessado nesse bloco, e coloca a informação sobre o Helper e o bloco em fila de espera para transferir. Assim que a conexão entre ambos estiver livre, o Collector pedirá o bloco ao Helper, que lho facultará sem pedir nada em troca. O Collector mantém uma fila de espera para cada Helper, e tenta sempre estar a receber de cada conexão um bloco.

4.5.2 Alternativa testada: Helpers colectivos

Como é bem conhecido, uma das formas de obter melhor desempenho num sistema BitTorrent é aumentar o número de participantes em modo de semente. Inspirada no protocolo 2Fast, que tem entidades privadas que difundem livremente blocos sem pedir nada em troca, os Helpers, e tendo em mente o objectivo de injectar sementes rapidamente no sistema, nasceu a ideia de criar entidades semelhantes aos Helpers, mas neste caso públicas. Deixa assim de existir o papel de Collector, pois os interessados são todos os participantes. Isto traz também alterações ao nível dos Helpers. Deixam de existir as restrições no que toca aos blocos em que estão interessados, passam a ser todos. No fundo esta nova entidade é relativamente semelhante a um participante normal, mas com uma nova imposição: não vai transferir todo o conteúdo, caso contrário seria em tudo igual aos restantes participantes.

Assim nasceu a entidade de Helper colectivo. Novamente, utilizadores com largura de banda disponível poderiam assumir este novo papel, transferindo uma porção de conteúdo em que não estão interessados de forma rápida, para depois permanecer na rede de partilha e simplesmente difundir os blocos que obtiveram sem pedir nada em troca.

A implementação deste novo papel foi feita, tal como no caso do protocolo 2Fast, estendendo as classes BTSession, BTDocument e BTAlgorithmPieceSelection, colocando a restrição ao interesse por qualquer bloco assim que determinada percentagem da transferência era alcançada, permanecendo a responder a pedidos de outros participantes, tal como uma semente, embora incompleta. As alterações foram substancialmente menos complexas que o protocolo 2Fast, com modificações mais breves ao algoritmo original do BitTorrent. No essencial, o

Helper colectivo continua a ser um participante normal da rede, até ao ponto em que deixa de estar interessado em transferir mais blocos do ficheiro, permanecendo como semente dos que entretanto conseguiu transferir.

4.5.3 Protocolo FairTorrent

Tal como abordado na secção 3.8, Alex Sherman et al. propõem uma alteração simples ao algoritmo de *Choking* do protocolo BitTorrent, que apresenta muito bons resultados do ponto de vista do desempenho. Como tal foi decidido implementar essas alterações para ver em que medida o desempenho da rede é melhorado.

A descrição do protocolo em maior pormenor pode ser encontrada na secção 3.8 ou em [16], e de forma muito resumida consiste em substituir o algoritmo de *Choking*. O novo algoritmo baseia-se numa lista ordenada por ordem crescente da dívida, em *bytes*, contraída face a cada participante conhecido pelo nó. Mantendo todas as ligações em estado *unchoked* e apenas respondendo aos pedidos pela ordem da lista, e apenas até saturar a ligação, tornam-se as trocas entre os participantes mais justas, e ao mesmo tempo melhora-se o desempenho global do sistema. A lista é reordenada a cada bloco enviado.

No campo da implementação, ao contrário do protocolo 2Fast, as alterações a efectuar foram mais pequenas, e mais contidas. Foi no entanto necessário proceder a algumas adaptações devido a limitações do simulador. Segue-se uma listagem das intervenções efectuadas:

- Para efeitos de dívida não são contabilizados *bytes* mas sim sub-blocos. Isto prende-se com o facto da contabilização de *bytes* no simulador ser mais complexa, uma vez que a sua contabilização tinha de ser recalculada sempre que há um reajustar da largura de banda de uma ligação e os tempos de envios dos sub-blocos são revistos, uma vez que os novos cálculos são efectuados com divisão entre os que teoricamente já teriam sido recebidos e os que estão por receber. Desta forma, contabilizando apenas os sub-blocos enviados e recebidos, não existe sequer grande diferença, uma vez que apenas os sub-blocos do último bloco podem tomar tamanhos diferentes.

- Foram acrescentados métodos para contabilizar os sub-blocos como dívida positiva ou negativa, no envio e recepção de blocos respectivamente.

- Na classe BTSession foi criada uma nova estrutura para gerir as dívidas para cada participante.

- Na classe BTAlgorithmReChoking foi necessário criar uma nova classe de ordenação dos participantes no *rechoke*, tendo agora em conta a dívida contraída para cada participante.

- Infelizmente, a forma como o simulador foi implementado não permite ter todas as ligações em modo *unchoke* e guardar pedidos para responder mais tarde, pois são precisamente os eventos de alteração do estado de uma ligação (*choked*/*unchoked*) que despoletam esses mesmos pedidos e consequentes envios de blocos. Teve então de existir uma adaptação, que mantém todas as ligações em modo *choked* como no BitTorrent e procede com as habituais rondas em períodos de 10 segundos, onde é reordenada a lista de participantes. Adicionalmente, o número de ligações simultâneas foi aumentado para 6 para, de alguma forma tentar rentabilizar este método alternativo.

- As sementes não se comportam no simulador da mesma forma que é descrita pelos autores do protocolo FairTorrent, mantendo as suas funcionalidades originais do protocolo BitTorrent. Esta alteração não foi feita em parte devido ao facto da alteração descrita no ponto anterior não ter sido efectuada, o que tornaria uma implementação das sementes muito complexa.

4.5.4 Alternativa testada: Trusted Peers

Mais uma vez com o objectivo de aumentar o número de sementes no sistema rapidamente, surgiu uma nova ideia. Partindo da visão de um mundo ideal, em que seria possível adivinhar quais os participantes que iriam ficar ligados em modo semente quando terminassem a transferência, faria todo o sentido dar prioridade a estes participantes, de modo a criar sementes mais rapidamente e aumentar assim o desempenho do sistema e minimizando os tempos de transferência de todos os participantes na rede de partilha.

Partindo da alteração feita ao algoritmo de *rechoke* pelo protocolo FairTorrent é possível atribuir facilmente um peso aos participantes que se sabe à partida que vão ficar em modo semente, dando-lhes assim prioridade e diminuindo os seus tempos de transferência.

Mas como é possível saber quais os participantes que vão de facto permanecer na rede após a transferência? Através de um registo de transferências passadas e calculando o rácio de cada utilizador. Isto apenas é possível em comunidades fechadas, com o registo de utilizadores e uma contagem segura dos volumes de dados transferidos ao longo do tempo. Efectuando o cálculo do rácio global de cada utilizador seria possível traçar um perfil de utilização: rácios inferiores a 1

evidenciam utilizadores que não estão interessados em permanecer ligados por muito tempo na rede, enquanto rácios superiores a 1 evidenciam utilizadores com um comportamento altruísta, que geralmente permanecem ligados em modo semente após concluir a transferência, tanto mais tempo quanto maior que 1 for o seu rácio.

Teríamos assim uma forma de poder afirmar, ainda que com alguma margem de erro, quais os participantes que irão permanecer ligados em modo de semente ou não. Estes serão designados de Trusted Peers, devido à confiança que neles é depositada, sendo-lhes oferecida prioridade na transferência com a promessa que quando terminarem a mesma se manterão na rede em modo de semente. Estão assim criadas as bases para um sistema potencialmente mais justo, e existe campo para a aplicação de uma política de incentivos: os utilizadores interessados em transferências mais rápidas são incentivados a permanecer ligados em modo de semente com o objectivo de atingir e manter o estatuto de Trusted Peers.

As alterações necessárias para testar esta hipótese foram realizadas sobre as efectuadas anteriormente no âmbito do protocolo FairTorrent. Apenas é feita uma simples intervenção na nova função de ordenação dos participantes, definida no BTAlgorithmReChoking. Nesta implementação, o parâmetro de ordenação é dividido em dois campos: a dívida obtida, e o facto de ser Trusted Peer ou não. O peso dado a cada um destes campos é gerido através de uma componente adicional, $1-\alpha$ e α, em cada um dos campos respectivamente. Variando α podemos alterar o peso que é atribuído aos Trusted Peers. Nesta implementação esse valor é dado por uma constante, definida globalmente, e que pode ser alterada para cada teste.

Para a implementação deste sistema parte-se da premissa que os participantes têm acesso a uma entidade central capaz de lhes afirmar com segurança se um outro participante é Trusted Peer ou não, assim como dar informações relativas ao rácio global desse participante, ou quaisquer outras que sejam necessárias.

Por último, de forma a poder transformar o facto de se ser Trusted Peer ou não num valor que possa ser comparado e equiparado à dívida para os participantes, a resposta da entidade central ditará a fórmula final a ser usada no cálculo da dívida a esse participante: se for Trusted Peer terá um peso igual à maior dívida conhecida, caso contrário será igual à menor.

Sendo V_i o valor da dívida ponderada para o participante i, este é dado pela fórmula

$$\begin{cases} V_i = ((1-\alpha) * D_i) + (\alpha * Max(D)) & \text{se } i \text{ é Trusted Peer} \\ V_i = ((1-\alpha) * D_i) + (\alpha * Min(D)) & \text{caso contrário} \end{cases} \tag{4.1}$$

Onde D_i é a dívida para o participante i e $Max(D)$ e $Min(D)$ as maior e menor dívidas entre todos os participantes conhecidos.

Esta distinção nas fórmulas a usar permite que os Trusted Peers tenham um valor final na mesma ordem de grandeza e normalmente equiparado aos participantes a quem se detém uma maior dívida. Desta forma aumenta-se a prioridade dos Trusted Peers, mas não se invalida por completo o envio de blocos a participantes normais.

À parte desta alteração mantêm-se todas as outras efectuadas no protocolo FairTorrent. Este algoritmo, que está ainda na sua primeira versão, tem potencial para poder sofrer alterações no sentido de o tornar mais adaptável, com recurso ao valor do rácio em vez de um simples booleano que define se um participante é Trusted Peer ou não.

No próximo capítulo apresenta-se de que formas foram testadas as diferentes variantes do algoritmo do BitTorrent que foram descritas.

5 . Estudos comparativos

5.1 Introdução

Neste capítulo serão abordados os testes efectuados sobre as variantes do protocolo implementadas. Para esse efeito foram traçados cenários para cobrir as mais diversas situações e conduzidos diversos testes no simulador, que permitiram tirar conclusões sobre os sistemas estudados, bem sobre qual o melhor rumo a tomar tendo em conta os objectivos deste trabalho.

Na secção 5.2 serão abordados os protocolos e alterações em estudo e na secção 5.3 descreve-se o ambiente de simulação e os cenários traçados. Na secção 5.4 apresentam-se e discutem-se os resultados dos testes e conclui-se na secção 5.5.

5.2 Protocolos testados

Além do protocolo BitTorrent, presente no simulador, foram implementados outros protocolos, e testados para comparar o seu desempenho. Segue-se uma breve descrição dos mesmos:

- **Protocolo 2FAST:** Foi usada a implementação descrita na secção 4.5.1 do protocolo 2Fast. Em resumo, o seu funcionamento passa pela existência de 2 novos papéis na rede de partilha: participantes dispostos a ajudar outros na conclusão da sua transferência, os Helpers, e participantes que pedem a ajuda de outros para concluir a sua própria transferência, os Collectores. Os Collectors contactam Helpers, que se tornam seus *proxies* na rede de partilha, obtendo os blocos que o Collector ainda não possui, enviando-os sem pedir nada em troca. Um Collector pode ter o número de Helpers que pretender. Para efeitos destes testes cada Collector está a usar 4 Helpers, um número razoável e que produz bons resultados. Assume-se também que os Helpers são clientes que não estando a efectuar transferências oferecem a sua largura de banda a outros, e como tal estão disponíveis logo desde o início da transferência para ajudar Collectors.

- **Helpers colectivos:** Com o objectivo de testar a hipótese descrita na secção 4.5.2, esta solução foi também submetida a testes. O seu funcionamento passa por criar nós, com o nome de Helpers colectivos, que irão entrar na rede de partilha, mesmo sem ter interesse no conteúdo, e transferir determinada percentagem do ficheiro com o objectivo de

oferecer os blocos que conseguiram transferir a todos os outros, actuando como sementes incompletas.

- **Protocolo FairTorrent:** De forma a testar o impacto que as alterações descritas na secção 4.5.3 têm no desempenho de uma rede de partilha, estas foram implementadas e testadas. Essencialmente o protocolo FairTorrent consiste na alteração da escolha de parceiros no algoritmo de *Choking*: em vez de escolher os parceiros pela taxa de *upload* que recebe dos mesmos, o participante escolhe responder aos pedidos dos parceiros a quem deve mais blocos.

- **Trusted Peers:** Trata-se de uma nova solução, baseada no protocolo FairTorrent e descrita em pormenor na secção 4.5.4. Através das observações feitas é possível concluir que a existência de mais sementes no sistema leva a melhores desempenhos do mesmo, e quanto mais cedo surgirem melhor. Assim, introduz-se uma alteração na ordenação dos participantes a quem enviar blocos, adicionando um novo elemento: o facto de o utilizador ser considerado Trusted Peer. Trusted Peers são os participantes que têm um comportamento maioritariamente altruísta e com grande probabilidade irão permanecer ligados em modo de semente após concluírem a transferência.

O teste destas diferentes soluções permite tirar conclusões sobre a viabilidade do seu uso, ou não, em redes reais, e qual o seu contributo para melhorar o desempenho de um sistema.

5.3 Cenários de testes

Os testes foram realizados numa mesma rede com 252 nós, dos quais apenas um determinado número participava na rede de partilha. Todas as ligações dos nós à rede eram simétricas, tinham a largura de banda de 1 MBPS e contavam com um *delay* de 15 ms. As ligações entre os nós de rede eram igualmente simétricas, tinham a largura de banda de 1000 MBPS e um *delay* de 5 ms. Todas as redes partilharam o mesmo ficheiro, com o tamanho de 100 MB. A entrada dos participantes da rede foi feita de forma faseada, respeitando uma aproximação de *flash crowd*, excepto os Helpers do cenário 2, que entram todos no instante inicial. Todos os testes foram repetidos 5 vezes, e foi considerada apenas a média de todos os resultados para efeitos de conclusões. Os restantes parâmetros de simulação variam conforme os cenários e serão descritos de seguida.

Foram criados 6 cenários de modo a tentar verificar qual o impacto de cada alteração nos tempos de transferência:

- **Cenário 1:** Cenário de controlo. Trata-se de uma rede de partilha bastante simples, contando com 5 sementes e 40 participantes, perfazendo o total de 45 intervenientes na rede de partilha.

- **Cenário 2:** Teste do protocolo 2Fast. Este cenário conta com 5 sementes, 35 participantes, 5 Collectors e 20 Helpers, perfazendo um total de 65 intervenientes na rede de partilha. Sem contabilizar os Helpers esta rede de partilha tem a mesma dimensão que o cenário 1, e contabilizado, a mesma dimensão do cenário 3.

- **Cenário 3:** Cenário de controlo com mais participantes, criado com o objectivo de ter uma rede de partilha com a mesma dimensão do cenário 2. Conta com 5 sementes e 60 participantes, perfazendo 65 intervenientes na rede de partilha.

- **Cenário 4:** Cenário de teste do conceito de Helpers colectivos. Este cenário conta com 5 sementes, 40 participantes e 20 Helpers colectivos, perfazendo 65 intervenientes na rede de partilha. Tal como o cenário 2 pode ser comparado com os cenários de controlo conforme sejam considerados os Helpers colectivos ou não. Neste cenário os participantes abandonam a rede de partilha quando terminam a transferência. Este cenário conta ainda com um parâmetro adicional: a percentagem do ficheiro que os Helpers colectivos obtêm, o qual toma os valores de 20%, 40%, 60%, 80% e 100%.

- **Cenário 5:** Este cenário foi criado com o objectivo de avaliar qual o impacto que as sementes têm de facto nos tempos de transferência. Assim, contém o dobro das sementes dos outros cenários: 10 sementes para 60 participantes. Este cenário é equivalente ao cenário 3, mas com mais sementes

- **Cenário 6:** Cenário de teste da alteração ao FairTorrent, os Trusted Peers. Conta com 60 participantes, dos quais 20 são considerados Trusted Peers. Os Trusted Peers foram escolhidos de forma a respeitarem a entrada faseada de participantes. Conta igualmente com 5 sementes iniciais. Após a conclusão da transferência, os participantes abandonam a rede de partilha à excepção dos Trusted Peers, que permanecem ligados em modo semente. Tal como o cenário 4, este cenário conta com um parâmetro adicional: o peso

dado à componente de um participante ser considerado Trusted Peer no cálculo da dívida, peso esse que toma os seguintes valores: 25%, 33%, 50%, 66% e 75%.

Os cenários 1,2 e 3 foram repetidos com algumas alterações de forma a poder tirar algumas conclusões adicionais. Em primeiro lugar foram criados os cenários 1+, 2+ e 3+ que representam os mesmos cenários mas com o dobro dos intervenientes excepto em relação às sementes que se mantêm 5. A título de exemplo, o cenário 2+ conta com 5 sementes, 70 participantes $(2*35)$, 10 Collectos $(2*5)$ e 40 Helpers $(2*20)$. Outro parâmetro variado é o comportamento dos participantes e Collectors após concluída a transferência: ou permanecem ligados na rede como sementes ou abandonam assim que concluída a transferência. Esta variação também foi aplicada ao cenário 5. Para efeitos de nomenclatura dos diferentes testes, S0 representa um cenário onde o comportamento final é abandonar a rede, e S1 manter-se ligado como semente.

Como forma de testar o protocolo FairTorrent o cenário 3 foi repetido, desta vez com este protocolo. Estes resultados são apresentados sob a denominação de CF e multiplicam-se conforme o comportamento após transferência.

Estes parâmetros adicionais vêm multiplicar os cenários base, obtendo-se assim 26 cenários em vez de 6. A tabela 5.1 resume os dados dos diferentes cenários de forma simples e de leitura fácil. O número na coluna dos *Helpers* representa também os Helpers colectivos do cenário 4 e na coluna *Parâmetro auxiliar* são mostrados os valores usados nos parâmetros auxiliares abordados na descrição dos cenários 4 e 6.

5.4 Resultados dos testes

Os resultados obtidos nos diversos cenários de teste permitem tirar várias conclusões, quer sobre o protocolo original do BitTorrent, quer sobre o impacto das alterações testadas. Estas conclusões são apresentadas nas próximas secções. Nas diversas tabelas que se seguem são apresentados tempos de transferência em segundos, seguidos de um valor entre parêntesis. Esse segundo valor representa o tempo quando comparado com o tempo teórico de transferência cliente - servidor baseado em linhas com as mesmas condições e com um nó cliente e um nó servidor, isto é, sem qualquer espécie de contenção. Este tempo teórico de transferência é considerado, naturalmente, o valor óptimo. Nos resultados comparando o número médio de blocos, o valor entre parêntesis representa a percentagem que esses mesmos blocos representam face ao número total de blocos que compõem o conteúdo a transferir. Um outro valor apresentado toma

Nome	Sementes	Participantes	Collectors	Helpers	Semente no final	Parâmetro auxiliar
C1S0	5	40	0	0	Não	—
C1S1	5	40	0	0	Sim	—
C1+S0	5	80	0	0	Não	—
C1+S1	5	80	0	0	Sim	—
C2S0	5	35	5	20	Não	—
C2S1	5	35	5	20	Sim	—
C2+S0	5	70	10	40	Não	—
C2+S1	5	70	10	40	Sim	—
C3S0	5	60	0	0	Não	—
C3S1	5	60	0	0	Sim	—
C3+S0	5	120	0	0	Não	—
C3+S1	5	120	0	0	Sim	—
C4-20	5	40	0	20	Não	20%
C4-40	5	40	0	20	Não	40%
C4-60	5	40	0	20	Não	60%
C4-80	5	40	0	20	Não	80%
C4-100	5	40	0	20	Não	100%
C5S0	10	60	0	0	Não	—
C5S1	10	60	0	0	Sim	—
CFS0	5	60	0	0	Não	—
CFS1	5	60	0	0	Sim	—
C6-25	5	60	0	0	—	25%
C6-33	5	60	0	0	—	33%
C6-50	5	60	0	0	—	50%
C6-66	5	60	0	0	—	66%
C6-75	5	60	0	0	—	75%

Tabela 5.1 Resumo dos cenários de teste

a denominação de rácio. O rácio é a medida que pode ser obtida dividindo o número de blocos oferecidos pelo número de blocos recebidos. Utilizadores com rácios superiores a 1 ofereceram mais blocos que receberam e utilizadores com rácio inferior a 1 receberam mais blocos do que ofereceram. Este valor permite uma leitura rápida e simples do nível de participação do utilizador.

5.4.1 Protocolo BitTorrent

Estudando os resultados obtidos nos cenários C3S0, C3S1, C5S0 e C5S1 podemos verificar o enorme impacto que o comportamento de *seeding* tem nas redes de partilha. A tabela 5.2 apresenta os tempos médios de transferência, bem como o valor mais rápido e mais lento registados nos cenários referidos. Os cenários contam com o mesmo número de utilizadores interessados no ficheiro, diferindo apenas no número de sementes (5 no cenário 3 e 10 no cenário 5) e no comportamento após a transferência.

Cenário	Tempo médio (seg.)	Mais rápido (seg.)	Mais lento (seg.)
C3S0	1595,8 (194,8%)	1060,2 (129,42%)	2957,7 (361,05%)
C5S0	1558 (190,19%)	944,2 (115,26%)	3200 (390,63%)
C3S1	1334,4 (162,89%)	1060,2 (129,42%)	1620,7 (197,84%)
C5S1	1240 (151,36%)	939,1 (114,63%)	1529,5 (187,71%)

Tabela 5.2 Tempos de transferência dos cenários 3 e 5

As diferenças entre os cenários com diferentes comportamentos são de cerca de 30% e 40% para os cenários 3 e 5, respectivamente, um número bastante revelador da importância que tem manter os utilizadores ligados em modo de semente quando terminam a transferência. Observando os tempos dos utilizadores mais rápidos não se notam grandes diferenças, mas nos mais lentos a diferença aproxima-se dos 200%, revelando onde está a falha. Ao abandonarem a rede assim que concluída a transferência, estes utilizadores estão a dificultar a obtenção de blocos a todos os outros, e a obrigar todos os seus anteriores parceiros a procurarem novas ligações, normalmente piores do que as que mantinham com eles. É também possível ver que aumentando o número de sementes iniciais se conseguem melhoramentos significativos.

Com estes dados podemos concluir que, como seria esperado, o número de sementes iniciais tem impacto no tempo de transferência do ficheiro por parte de todos os participantes, mas é o comportamento que os utilizadores têm quando terminam a transferência do conteúdo que tem mais impacto no desempenho do sistema. Torna-se assim necessário criar incentivos para

que os utilizadores permaneçam no sistema durante mais tempo em modo de semente, com vista a aumentar o desempenho do sistema e a diminuir o tempo médio de transferência dos utilizadores.

Um outro aspecto interessante referente ao protocolo BitTorrent é a escalabilidade. É possível ver que quanto maior o sistema, menos é exigido às sementes iniciais, sendo o peso repartido por todo os participantes. O custo deste comportamento traduz-se em tempos de transferência piores quanto maior a rede de partilha. A tabela 5.3 mostra diversos cenários, onde o comportamento final é variável, qual a dimensão da rede de partilha, qual o número de blocos que as sementes ofereceram, em média, o rácio obtido pelos utilizadores, também em média, e o tempo médio de transferência, em segundos.

Cenário	Dimensão da rede	Blocos oferecidos pelas sementes	Rácio dos utilizadores	Tempo médio (seg.)
C1S0	45	696,52 (178,14%)	0,78	1575,4 (192,31%)
C3S0	65	652,04 (166,76%)	0,86	1595,8 (194,8%)
C1+S0	85	755,12 (193,13%)	0,88	1616,7 (197,35%)
C3+S0	125	596,32 (152,51%)	0,94	1657,1 (202,28%)
C1S1	45	530,44 (135,66%)	0,83	1282,7 (156,58%)
C3S1	65	516,84 (132,18%)	0,89	1334,4 (162,89%)
C1+S1	85	547,16 (139,94%)	0,91	1334,4 (162,89%)
C3+S1	125	465,56 (119,07%)	0,95	1394 (170,17%)

Tabela 5.3 Rácios e tempos médios de transferência dos cenários 1, 1+, 3 e 3+

Observa-se que, excepto nos cenários C1+S0 e C1+S1, os blocos oferecidos pelas sementes tendem a decrescer, e o rácio dos utilizadores a subir, aproximando-se de 1. Este comportamento pode ser explicado pelo peso que as sementes representam no sistema ir diminuindo. Nos primeiros cenários trata-se de 5/45 enquanto nos últimos 5/125. Demonstra-se assim como escala o protocolo BitTorrent: o que não é oferecido pelas sementes é oferecido pelos restantes utilizadores da rede, e quantos mais utilizadores maior a oferta e menos se exige das sementes.

Quanto aos tempos médios de transferência, verifica-se que estes tendem a aumentar com o número de participantes na rede de partilha, embora as diferenças não sejam muito significativas. Este agravamento deve-se, provavelmente, à dificuldade em conseguir alcançar as sementes ou os participantes que detêm os blocos mais raros na rede de partilha. Previsivelmente os tempos de transferência são muito mais baixos nos cenários em que o comportamento dos utilizadores após a transferência é permanecer como sementes, oferecerendo a outros todos os blocos que estes necessitem.

É também possível ver que os cenários em que o comportamento dos utilizadores é permanecerem ligados como sementes após concluída a transferência são também aqueles que exigem menos das sementes, e onde se obtém melhores tempos de transferência. Este fenómeno deve-se ao facto de no final a oferta de blocos ser o mais diversificada possível, com todos os utilizadores que terminaram presentes no sistema como sementes.

5.4.2 Protocolo 2Fast

Os resultados obtidos pelo protocolo 2Fast podem ser consultados na tabela 5.4, onde são comparados com os resultados dos cenários 1 e 3, tanto nos cenários onde os utilizadores se mantinham ligados após a transferência como os cenários em que abandonavam a rede após transferência.

Cenário	Tempo médio (seg.)	Mais rápido (seg.)	Mais lento (seg.)
C1S0	1575,4 (192,31%)	1007,6 (123%)	3129 (381,96%)
C1S1	1282,7 (156,58%)	1007,6 (123%)	1518,7 (185,39%)
C2S0 - Utilizadoes	1632 (199,22%)	982,9 (119,99%)	6632,5 (809,2%)
C2S0 - Collectors	964,2 (117,7%)	853,9 (104,24%)	1068,4 (130,42%)
C2S1 - Utilizadores	1166,1 (142,34%)	952,7 (116,29%)	1424,7 (173,92%)
C2S1 - Collectors	896,1 (109,38%)	757,8 (92,51%)	1004,8 (122,66%)
C3S0	1595,8 (194,8%)	1060,2 (129,42%)	2957,7 (361,05%)
C3S1	1334,4 (162,89%)	1060,2 (129,42%)	1620,7 (197,84%)

Tabela 5.4 Tempos de transferência dos cenários 1, 2 e 3

Em primeiro lugar observa-se que os tempos médios de transferência obtidos pelos Collectors são inferiores aos obtidos pelos outros utilizadores, o que vem demonstrar a eficácia do protocolo. Nota-se também que a diferença de desempenho é maior no cenário com maior escassez de blocos, onde os utilizadores abandonam a rede quando terminam a transferência, demonstrando assim que o protocolo produz maiores ganhos quanto piores as condições de rede. O valor inferior a 100% obtido no tempo mais rápido de transferência dos Collectos no cenário onde os utilizadores se mantêm na rede em modo de semente deve-se provavelmente a erros de aproximações no simulador. Assume-se que este valor é obtido devido a um uso muito próximo dos 100% das ligações de rede deste nó, algo apenas alcançado devido ao bom desempenho do protocolo.

Quando comparados com os outros cenários, os valores dos utilizadores mantêm-se relativamente próximos, sem grandes diferenças, o que vem de alguma forma validar os resultados alcançados. Nota-se no entanto algum agravamento dos tempos no cenário em que os participantes abandonam após transferência, e um melhoramento nos tempos do cenário em que os utilizadores se mantém ligados em modo semente.

Cenário	Participante	Rácio	Blocos recebidos	Blocos oferecidos
C1S0	Sementes	∞	0 (0%)	696,52 (178,14%)
	Utilizadores	0,78	391 (100%)	303,93 (77,73%)
	Collectors	—	—	—
	Helpers	—	—	—
C1S1	Sementes	∞	0 (0%)	530,44 (135,66%)
	Utilizadores	0,83	391 (100%)	324,7 (83,04%)
	Collectors	—	—	—
	Helpers	—	—	—
C2S0	Sementes	∞	0 (0%)	624,88 (159,82%)
	Utilizadores	0,64	391 (100%)	248,39 (63,53%)
	Collectors	1,14	280,28 (71,68%)	318,56 (81,47%)
	Helpers	1,32	259,21 (66,29%)	339,52 (86,83%)
C2S1	Sementes	∞	0 (0%)	492,56 (125,97%)
	Utilizadores	0,68	391 (100%)	265,89 (68%)
	Collectors	1,43	267,16 (68,33%)	382,04 (97,71%)
	Helpers	1,27	246,85 (63,13%)	313,89 (80,28%)
C3S0	Sementes	∞	0 (0%)	652,04 (166,76%)
	Utilizadores	0,86	391 (100%)	336,67 (86,1%)
	Collectors	—	—	—
	Helpers	—	—	—
C3S1	Sementes	∞	0 (0%)	516,84 (132,18%)
	Utilizadores	0,89	391 (100%)	347,93 (88,98%)
	Collectors	—	—	—
	Helpers	—	—	—

Tabela 5.5 Rácios de cada classe de participantes nos cenários 1, 2 e 3

Na tabela 5.5 são apresentados os dados referentes aos rácios, blocos recebidos e blocos oferecidos em média por cada tipo de participante na rede de partilha. Apenas são contabilizados blocos trocados através do protocolo BitTorrent, os blocos trocados entre Collectors e Helpers não se encontram nesta tabela, mas podem ser calculados facilmente, pois trata-se do restante para alcançar os 100% do ficheiro, uma vez que todos os Collectors terminaram

efectivamente a transferência do ficheiro.

Apesar do tempo médio de transferência dos utilizadores ser semelhante entre todos os cenários, nota-se uma grande diferença no rácio, sendo muito inferior no cenário 2. Observando o valor dos blocos oferecidos pelas sementes, nota-se também uma diminuição. Ou seja, no cenário 2 são exigidos menos blocos das sementes e dos utilizadores. Esses blocos são então fornecidos por Collectors e Helpers, que registam, dentro do algoritmo BitTorrent, mais blocos oferecidos que os recebidos. O facto de os Collectors obterem blocos fora do algoritmo BitTorrent acelera-lhes muito a transferência, tornando-os sementes mais rapidamente nos cenários em que os participantes se mantêm ligados em modo semente, e retira-os da rede muito rapidamente nos cenários em que os participantes abandonam a rede, o que explica os tempos superiores e inferiores, obtidos pelos utilizadores nos cenários onde os participantes ficam em modo semente, ou não, respectivamente.

Em cenários com abundância de blocos, onde os utilizadores se mantêm ligados em modo semente, a existência de Collectors e Helpers vem beneficiar o desempenho do sistema, mas em cenários com escassez de recursos, apesar de em teoria também serem benéficos (o rácio superior a 1 tanto de Collectors como Helpers indica que ofereceram mais blocos à rede do que aqueles que receberam até ao momento que abandonaram) observa-se um agravamento dos tempos médios de transferência.

Cenário	Tempo médio (seg.)	Mais rápido (seg.)	Mais lento (seg.)
C1+S0	1616,7 (197,35%)	1116,3 (136,26%)	3200,4 (390,67%)
C1+S1	1334,4 (162,89%)	1114,4 (136,04%)	1457,9 (177,97%)
C2+S0 - Utilizadoes	1830,7 (223,48%)	1060,5 (129,46%)	9460,5 (1154,85%)
C2+S0 - Collectors	1046,3 (127,73%)	930,7 (113,61%)	1131,7 (138,14%)
C2+S1 - Utilizadores	1215,4 (148,37%)	1035,8 (126,44%)	1528,5 (186,58%)
C2+S1 - Collectors	1019,2 (124,42%)	832,4 (101,61%)	1167,3 (142,49%)
C3+S0	1657,1 (202,28%)	1029,9 (125,71%)	4049 (494,26%)
C3+S1	1394 (170,17%)	1029,8 (125,71%)	1623,8 (198,22%)

Tabela 5.6 Tempos de transferência dos cenários 1+, 2+ e 3+

Nos cenários correspondentes aos 3 abordados anteriormente, mas com o dobro dos intervenientes (cenários 1+, 2+ e 3+), registaram-se os valores para os tempos de transferência apresentados na tabela 5.6. Aqui as conclusões tiradas anteriormente confirmam-se, existindo de facto um acentuar das mesmas, tanto do ganho de desempenho de Collectors face aos restantes

utilizadores, como das diferenças nos tempos médios de transferência dos utilizadores nos diferentes cenários. O valor do Collector mais rápido encontra-se desta vez perto do valor real, demonstrando a total utilização da rede para efectuar a transferência num tempo muito próximo do tempo necessário para transferir o conteúdo a partir de um servidor dedicado e sem concorrência.

Cenário	Participante	Rácio	Blocos recebidos	Blocos oferecidos
C1+S0	Sementes	∞	0 (0%)	755,12 (193,13%)
	Utilizadores	0,88	391 (100%)	343,81 (87,93%)
	Collectors	—	—	—
	Helpers	—	—	—
C1+S1	Sementes	∞	0 (0%)	547,16 (139,94%)
	Utilizadores	0,91	391 (100%)	356,8 (91,25%)
	Collectors	—	—	—
	Helpers	—	—	—
C2+S0	Sementes	∞	0 (0%)	546,72 (139,83%)
	Utilizadores	0,72	391 (100%)	280,45 (71,73%)
	Collectors	1,32	277,1 (70,87%)	366,72 (93,79%)
	Helpers	1,38	269,11 (68,83%)	371,84 (95,1%)
C2+S1	Sementes	∞	0 (0%)	458,24 (117,2%)
	Utilizadores	0,72	391 (100%)	287,82 (73,61%)
	Collectors	1,57	274,2 (70,13%)	429,96 (109,96%)
	Helpers	1,32	265,78 (67,98%)	350,13 (89,55%)
C3+S0	Sementes	∞	0 (0%)	596,32 (152,51%)
	Utilizadores	0,94	391 (100%)	366,15 (93,65%)
	Collectors	—	—	—
	Helpers	—	—	—
C3+S1	Sementes	∞	0 (0%)	465,56 (119,07%)
	Utilizadores	0,95	391 (100%)	371,6 (95,04%)
	Collectors	—	—	—
	Helpers	—	—	—

Tabela 5.7 Rácios de cada classe de participantes nos cenários 1+, 2+ e 3+

A tabela 5.7, que contém o rácio, os blocos recebidos e os oferecidos de cada tipo de participante vem também confirmar os resultados obtidos nos cenários anteriores, existindo poucas diferenças entre eles, excepto na quantidade de blocos oferecidos pelas sementes. Esta diferença de valores corresponde ao fenómeno que foi já descrito na secção 5.4.1, referente à escala do sistema: quanto maior, menos peso têm as sementes.

Desta forma, após análise dos resultados, conclui-se que o protocolo 2Fast funciona, melhorando o desempenho dos utilizadores que se tornam Collectors, podendo mesmo nos casos de abundância de blocos ser benéfico para o desempenho de todo o sistema. O papel de Collector pode ser aliciante para os utilizadores dispostos a trocar algo por transferências mais rápidas, e o papel de Helpers aliciante para os utilizadores com largura de banda disponível e que gostariam de ganhar algo com isso (directamente dos Collectors ou através de um sistema regulador do protocolo 2Fast). Consideramos assim que o protocolo 2Fast possuí de facto, como indicado pelos autores, as bases para que a sua utilização num sistema com trocas de serviços e/ou incentivos seja viável.

5.4.3 Helpers colectivos

Foram feitos testes em que os Helpers colectivos iniciavam a sua transferência como um nó normal mas paravam a mesma quando atingiam 20%, 40%, 60%, 80% e 100% da mesma. Os tempos obtidos pelos utilizadores são apresentados na tabela 5.8.

Cenário	Tempo médio (seg.)	Mais rápido (seg.)	Mais lento (seg.)
C1S0	1575,4 (192,31%)	1007,6 (123%)	3129 (381,96%)
C3S0	1595,8 (194,8%)	1060,2 (129,42%)	2957,7 (361,05%)
C4-20	1680,6 (205,16%)	1023,7 (124,96%)	4898,9 (598,01%)
C4-40	1574,8 (192,23%)	1031,1 (125,87%)	4024,9 (491,32%)
C4-60	1657,4 (202,31%)	1026,2 (125,27%)	4479,1 (546,76%)
C4-80	1630,7 (199,06%)	1043,1 (127,33%)	4508,8 (550,39%)
C4-100	1579,2 (192,77%)	1055,2 (128,81%)	2965,8 (362,04%)

Tabela 5.8 Tempos de transferência do cenário 4, com os tempos dos cenários 1 e 3 para controlo

Os tempos observados não permitem concluir sobre pontos óptimos ou progressões uma vez que não são monótonos e até relativamente próximos. Comparando-os com os valores obtidos nos cenários 1 e 3 é possível concluir que não só não introduzem melhorias no desempenho como o agravam ligeiramente.

Observando os valores obtidos para o rácio e para os blocos recebidos e enviados, que se encontram na tabela 5.9, constata-se como os Helpers colectivos falham definitivamente em oferecer uma possibilidade de rentabilizar de forma positiva a largura de banda de utilizadores desocupados. No cenário em que os Helpers param ao transferir 20% do ficheiro obtém-se um excelente rácio de 1,5, o que quer dizer que, em média, cada Helper está a oferecer mais 50% do

Cenário	Participante	Rácio	Blocos recebidos	Blocos oferecidos
C1S0	Sementes	∞	0 (0%)	696,52 (178,14%)
	Utilizadores	0,78	391 (100%)	303,93 (77,73%)
C3S0	Sementes	∞	0 (0%)	652,04 (166,76%)
	Utilizadores	0,86	391 (100%)	336,67 (86,1%)
C4-20	Sementes	∞	0 (0%)	681,16 (174,21%)
	Utilizadores	0,73	391 (100%)	285,09 (72,91%)
	Helpers	1,5	83,26 (21,29%)	124,79 (31,92%)
C4-40	Sementes	∞	0 (0%)	617,08 (157,82%)
	Utilizadores	0,75	391 (100%)	291,25 (74,49%)
	Helpers	1,27	161,62 (41,34%)	205,08 (52,45%)
C4-60	Sementes	∞	0 (0%)	607,6 (155,4%)
	Utilizadores	0,75	391 (100%)	295,16 (75,49%)
	Helpers	1,17	239,42 (61,23%)	279,2 (71,41%)
C4-80	Sementes	∞	0 (0%)	606,72 (155,17%)
	Utilizadores	0,79	391 (100%)	306,41 (78,37%)
	Helpers	1,05	316,4 (80,92%)	333,9 (85,4%)
C4-100	Sementes	∞	0 (0%)	649,44 (166,10%)
	Utilizadores	0,84	391 (100%)	327,2 (83,68%)
	Helpers	0,91	391 (100%)	356,24 (91,11%)

Tabela 5.9 Rácios de cada classe de participantes nos cenários 4, com os rácios dos cenários 1 e 3 para controlo

que aquilo que recebeu, o que se comprova na oferta de 30% de blocos. Infelizmente acabam por ser poucos blocos, não tendo grande impacto nos tempos médios de transferência dos utilizadores como já observado na tabela anterior. Quanto maior a percentagem do ficheiro que os Helpers obtêm, pior o rácio que obtêm, chegando a um rácio inferior a 1 no último cenário de 100%. Nota-se no entanto uma ligeira redução na quantidade de blocos que as sementes oferecem se os Helpers transferirem maior percentagem de blocos. Ou seja, é necessário que os Helpers transfiram mais blocos para que o seu impacto seja observado nas sementes, no entanto quanto maior a percentagem que transferem, pior é o rácio obtido. E tudo isto sem introduzir melhoramentos no desempenho do algoritmo.

Analisando em maior detalhe o último cenário, esclarece-se definitivamente que usar Helpers colectivos não é uma opção viável. Trata-se de um cenário composto por 5 sementes iniciais, 40 utilizadores e 20 Helpers colectivos, em que o comportamento dos utilizadores quando terminam a transferência é abandonar, enquanto os Helpers vão permanecer em modo semente. Observando os tempos médios de transferência para os utilizadores, o valor obtido neste cenário é 2% inferior ao cenário 3, com 60 utilizadores em que nenhum fica ligado em modo semente quando termina a transferência, e igual ao tempo registado no cenário 1, em que 40 utilizadores terminam a transferência, igualmente sem permanecerem ligados em modo semente ao concluir a transferência.

Assim, com os dados que dispomos, concluímos que os Helpers colectivos não trazem quaisquer benefícios à rede de partilha, e como tal a sua utilização por parte de redes baseadas em trocas de serviços e incentivos parece não merecer interesse.

5.4.4 Protocolo FairTorrent

Os resultados obtidos, e apresentados na tabela 5.10, vêm comprovar os já obtidos pelos autores em [16], demonstrando que as alterações introduzidas no algoritmo original do BitTorrent melhoram o desempenho global do sistema.

Cenário	Tempo médio (seg.)	Mais rápido (seg.)	Mais lento (seg.)
C3S0	1595,8 (194,8%)	1060,2 (129,42%)	2957,7 (361,05%)
CFS0	1402,2 (171,17%)	893,4 (109,05%)	2382,8 (290,87%)
C3S1	1334,4 (162,89%)	1060,2 (129,42%)	1620,7 (197,84%)
CFS1	1223,9 (149,4%)	893,4 (109,05%)	1477 (180,3%)

Tabela 5.10 Tempos de transferência do cenário 3 com e sem protocolo FairTorrent

Foi também registado, e tal pode verificar-se na tabela 5.11, que este protocolo exige mais recursos às sementes iniciais, o que resulta em rácios inferiores. Este fenómeno é mais acentuado no cenário onde os participantes abandonam a rede após concluir a transferência e pode ser explicado pelo facto dos participantes concluírem a transferência mais rápido, abandonando mais cedo a rede, e deixando os participantes mais lentos restritos a cada vez menos participantes para além das sementes, recaindo assim mais pedidos sobre as mesmas.

Cenário	Rácio dos utilizadores	Blocos oferecidos pelas sementes
C3S0	0,86	652,04 (166,76%)
CFS0	0,82	841,52 (215,22%)
C3S1	0,89	516,84 (132,18%)
CFS1	0,87	614,56 (157,18%)

Tabela 5.11 Rácio dos utilizadores e número de blocos oferecido pelas sementes no cenário 3, com e sem protocolo FairTorrent

As alterações introduzidas pelo FairTorrent são relativamente simples, mas permitem obter ganhos de desempenho significativos. Desta forma, o protocolo FairTorrent deverá ser considerado por qualquer sistema que pretenda melhorar o desempenho do protocolo BitTorrent. Com efeito, foi através deste protocolo que nasceu a ideia dos Trusted Peers, cujos resultados são abordados na próxima secção.

5.4.5 Trusted Peers

As alterações ao protocolo FairTorrent revelaram ganhos de desempenho, quer no global, quer nos tempos dos diferentes participantes, como pode ser observado na tabela 5.12, onde se encontram os tempos de transferência dos participantes. Adicionalmente encontram-se também na mesma tabela os tempos obtidos nos cenários referentes ao protocolo FairTorrent e o cenário de controlo com um número de participantes equivalente, o cenário 3.

Numa primeira análise é possível comprovar como globalmente os tempos médios de transferência dos cenários baseados em Trusted Peers são iguais ou melhores que o cenário CFS1, ou seja, o cenário do protocolo FairTorrent onde todos os participantes permanecem ligados como sementes após concluírem a transferência. Ou seja, os cenários baseados em Trusted Peers são capazes de alcançar tempos médios de transferência iguais ou melhores que os cenários baseados no protocolo FairTorrent, com apenas um terço das sementes finais.

Uma análise mais pormenorizada aos tempos médios obtidos pelos diferentes participantes

Cenário	Participante	Tempo médio (seg.)	Mais rápido (seg.)	Mais lento (seg.)
C3S0	Utilizadores	1595,8 (194,8%)	1060,2 (129,42%)	2957,7 (361,05%)
CFS0	Utilizadores	1402,2 (171,17%)	893,4 (109,05%)	2382,8 (290,87%)
C3S1	Utilizadores	1334,4 (162,89%)	1060,2 (129,42%)	1620,7 (197,84%)
CFS1	Utilizadores	1223,9 (149,4%)	893,4 (109,05%)	1477 (180,3%)
C6-25	Global	1227,2 (149,8%)	877 (107,05%)	1715,6 (209,42%)
	Utilizadores	1294,7 (158,05%)	877 (107,05%)	1715,6 (209,42%)
	Trusted Peers	1092,2 (133,32%)	974,7 (118,98%)	1288,1 (157,24%)
C6-33	Global	1218,9 (148,79%)	857 (104,61%)	1740,6 (212,48%)
	Utilizadores	1313,7 (160,36%)	857 (104,61%)	1740,6 (212,48%)
	Trusted Peers	1029,3 (125,64%)	925,2 (112,93%)	1310,2 (159,94%)
C6-50	Global	1205,6 (147,16%)	826,4 (100,88%)	1942,1 (237,07%)
	Utilizadores	1330,3 (162,39%)	882,9 (107,78%)	1942,1 (237,07%)
	Trusted Peers	956,1 (116,72%)	826,4 (100,88%)	1297,7 (158,41%)
C6-66	Global	1199,6 (146,44%)	843,3 (102,94%)	1891,8 (230,94%)
	Utilizadores	1327,9 (162,09%)	973,3 (106,97%)	1891,8 (230,94%)
	Trusted Peers	943,1 (115,13%)	843,3 (102,94%)	1140,8 (139,26%)
C6-75	Global	1201,6 (146,67%)	856,1 (104,5%)	1927,2 (235,25%)
	Utilizadores	1324,9 (161,73%)	887,3 (108,31%)	1927,2 (235,25%)
	Trusted Peers	954,9 (116,56%)	856,1 (104,5%)	1175,2 (143,46%)

Tabela 5.12 Tempos de transferência do cenário 6, com controlo feito pelo cenário 3 com e sem protocolo FairTorrent

permite ver o ganho real de cada um. Os utilizadores normais, que vão efectuar a sua transferência e abandonar a rede logo de seguida, obtêm tempos na ordem dos 160% do melhor tempo teoricamente possível. Isto traduz-se num ganho de desempenho de 6,4% se comparado com o cenário adequado, isto é, o cenário 3, onde os participantes abandonam a rede após concluída a transferência, cenário esse onde se regista um tempo médio de transferência de 171% do melhor tempo teoricamente possível.

No que toca aos Trusted Peers, os tempos registados variam conforme o peso dado ao facto de serem Trusted Peers nas trocas entre os participantes, e vão desde os 133% aos 117% do melhor tempo teoricamente possível. Comparando estes valores com o melhor obtido nos cenários do protocolo FairTorrent vê-se uma melhoria significativa de desempenho, entre os 10,7% e os 21,5%, conforme o peso.

Verifica-se assim que, independentemente do peso atribuído aos Trusted Peers nas trocas entre participantes, se obtêm ganhos de desempenho para todos os intervenientes no sistema, ao mesmo tempo que se cria um incentivo muito forte para que os participantes permaneçam no sistema como sementes: ganhar o estatuto de Trusted Peers para poder ter acesso a transferências mais rápidas.

Uma última análise aos tempos médios de transferência dos vários participantes evidência o funcionamento dos objectivos a que esta alteração se propôs: ao aumentar o peso que os Trusted Peers têm nas trocas entre participantes, consegue-se reduzir o tempo médio de transferência dos mesmos de 133% para 117%, enquanto o tempo médio de transferência dos restantes utilizadores apenas se agrava de 158% para 161%, quando comparados com o melhor valor obtido com a solução. Isto regista-se desta forma porque ao acelerar as transferências dos Trusted Peers se criam sementes mais depressa, que assim vão acelerar todo o sistema globalmente.

Cenário	Rácio dos utilizadores (seg.)	Rácio dos Trusted Peers	Rácio global	Blocos oferecidos pelas sementes
C3S0	—	—	0,86	652,04 (166,76%)
CFS0	—	—	0,82	841,52 (215,22%)
C3S1	—	—	0,89	516,84 (132,18%)
CFS1	—	—	0,87	614,56 (157,18%)
C6-25	0,69	1,19	0,87	650,2 (166,29%)
C6-33	0,65	1,28	0,86	647,36 (165,57%)
C6-50	0,63	1,34	0,87	619,88 (158,54%)
C6-66	0,63	1,35	0,87	604,36 (154,57%)
C6- 75	0,64	1,33	0,87	616,36 (157,64%)

Tabela 5.13 Rácio dos utilizadores e número de blocos oferecido pelas sementes no cenário 6, com controlo feito pelo cenário 3 com e sem protocolo FairTorrent

Analisando os valores dos rácios e dos blocos oferecidos pelas sementes iniciais, presentes na tabela 5.13, é possível tirar várias conclusões. Em primeiro lugar, o rácio global do sistema nos cenários baseados em Trusted Peers é relativamente constante e muito próximo ao cenário do protocolo FairTorrent em que os utilizadores permanecem como sementes após concluírem a transferência, mesmo contando com apenas um terço das sementes no final. Observa-se ainda que as sementes oferecem menos blocos quanto maior o peso dos Trusted Peers nas trocas entre participantes, ao mesmo tempo o rácio dos Trusted Peers aumenta e o dos utilizadores diminui ligeiramente. Isto vem mais uma vez comprovar que, ao dar mais peso aos Trusted Peers nas trocas entre participantes, estes terminam mais cedo a sua transferência e permanecem mais tempo como sementes no sistema, reduzindo assim o impacto que as sementes iniciais têm no mesmo.

Dados os resultados alcançados pode-se concluir que a utilização de Trusted Peers numa rede de partilha fechada, com utilizadores registados e registo de histórico de transferências, seria benéfica para a rede, mesmo no caso em que o comportamento típico dos utilizadores é abandonar o sistema após o fim das suas transferências. A sua introdução iria permitir alcançar melhorias de desempenho notórias ao mesmo tempo que criaria mecanismos de incentivos capazes de aliciar os utilizadores a permanecer no sistema como sementes, aliciados com os tempos de transferência mais baixos conseguidos pelos utilizadores com o estatuto de Trusted Peers. E a única forma de manter o estatuto seria manter o comportamento de permanecer ligado em modo semente quando concluídas as transferências. Seria ainda possível, em futuras implementações, usar uma variante da função que dita a dívida para um participante, e em vez de usar um valor fixo para o peso dos Trusted Peers, este ser dado pelo rácio, recompensando desta forma utilizadores com rácios diferentes de formas diferentes, criando um sistema de incentivos mais justo. Por outro lado, se o estatuto de Trusted Peer fosse também tomado em consideração pelas sementes, dando prioridade a estes participantes, poderíamos obter melhorias suplementares para os próprios e para a rede globalmente.

5.5 Conclusões

Neste capítulo foram realizados alguns testes sobre o protocolo BitTorrent para analisar as suas propriedades, e apresentados os resultados do estudo de alterações e alternativas ao mesmo, que poderiam trazer benefícios para redes baseadas em troca de serviços e/ou incentivos. Os

testes permitiram ainda tirar conclusões acerca das vantagens e desvantagens alcançadas por cada uma das propostas.

Uma primeira conclusão que sobressai é o impacto que o comportamento dos utilizadores após a conclusão da transferência tem para o desempenho global de um sistema baseado no protocolo BitTorrent. As conclusões são muito claras, o número de sementes e o comportamento dos participantes que permanecem no sistema como sementes após a transferência do conteúdo em que estavam interessados, permite influenciar de forma decisiva o desempenho da rede. Sabendo que o desempenho de um sistema passa pelo número de sementes nas redes de partilha, é importante desenvolver estratégias que resultem em incentivar as sementes a permanecer durante períodos maiores no sistema.

Confirmou-se o bom desempenho do protocolo 2Fast, que mostrou oferecer bom desempenho aos utilizadores dispostos a oferecer algo em troca de transferências mais rápidas, ao mesmo tempo que oferece aos utilizadores desocupados a possibilidade de tirar algo em troca de colocar a sua largura de banda ao serviço de outros, abrindo assim caminho para a criação de eventuais sistemas de incentivos, como os desenvolvidos pelos seus proponentes.

A alteração estudada que tinha por base criar sementes mais rapidamente no sistema, através da criação dos denominados Helpers colectivos, mostrou não trazer vantagens para a rede, e como tal o seu conceito foi abandonado, não tendo sido alvo de mais estudos.

A implementação do protocolo FairTorrent esteve à altura das melhorias de desempenho divulgadas pelos autores, mostrando ser capaz de melhorar o desempenho do sistema com trocas mais justas, mesmo sem recorrer a aumentos no número de sementes.

Foi com base no protocolo FairTorrent que surgiu a principal contribuição inovadora deste trabalho: os Trusted Peers. Os testes revelaram que adaptar o protocolo FairTorrent a uma política de reputação dos participantes atinge bons resultados no campo do desempenho, quer do sistema, quer das diferentes classes de utilizadores. Estamos assim perante uma situação em que todos os intervenientes têm algo a ganhar, e, a confirmarem-se estes resultados noutras ferramentas, capazes de modelar não só rede mas também o comportamento dos utilizadores de uma forma mais aproximada da realidade, seria interessante ver quais os resultados obtidos em redes reais, e se de facto uma política de incentivos associada ao protocolo FairTorrent, e baseada na reputação, poderia melhorar o desempenho da rede de forma global.

Os resultados aqui obtidos referem-se apenas a redes como as descritas na secção 5.3, ou seja, redes simétricas, onde o comportamento de cada utilizador é controlado. Em cenários reais, com linhas assimétricas e onde o comportamento de cada utilizador difere de inúmeras

formas, os resultados podem ser diferentes dos aqui apresentados. Não foi possível realizar mais testes no tempo disponível, no entanto os resultados obtidos com os testes apresentados são bastante promissores.

6. Análise de viabilidade da utilização da proposta: Trusted Peers

No capítulo anterior foram testadas as implementações introduzidas no simulador, e os resultados mostraram existir interesse na adopção da solução designada Trusted Peers. Neste capítulo será abordada a problemática da implementação e utilização dessa solução em ambientes reais, que pode ser dividida em vários aspectos. A secção 6.1 aborda o comportamento dos utilizadores e a secção 6.2 descreve a política de incentivos subjacente à solução. A secção 6.3 dá pistas sobre a implementação da solução em ambiente real e quais os aspectos que acarretam maior relevância, assim como apresenta uma breve discussão sobre questões de segurança da implementação. Finalmente, conclui-se na secção 6.4.

6.1 Comportamento dos utilizadores

Segundo observado por Matei Ripeanu et. al. em [14] a grande maioria dos utilizadores do protocolo BitTorrent, independentemente do cliente que usam, tem um comportamento que se baseia em obter o conteúdo que pretendem e abandonar a rede, como demonstrado pelos rácios médios obtidos entre os 30% e os 60%, sendo os valores mais altos registados em comunidades fechadas. Ou seja, a maioria dos utilizadores não devolve à rede sequer metade dos recursos que consumiu. Isto confima que os utilizadores comuns conseguem obter o que pretendem a troco de pouca participação e obtêm mais do que aquilo que contribuem.

Apesar de ser em menor escala, existem ainda utilizadores, chamados de *free-riders* cujo objectivo é fazer a transferência sem oferecer nada à rede de partilha, munindo-se de clientes estratégicos que actuam segundo esse mesmo propósito.

Esta realidade pode ser verificada por evidência empírica do comportamento típico dos utilizadores e só é compensada pelo facto de existir um outro grupo de utilizadores que, por uma razão ou outra, se disponibilizam como sementes a tempo inteiro. Este facto é comprovado em [14] quando se observa que, em média, 79% da carga de transferência numa rede de partilha recai sobre 10% dos utilizadores. Se este último grupo de utilizadores desaparecesse porque deixaram de ter motivação para o seu comportamento, o sistema provavelmente degradar-se-ia e perderia grande parte da sua popularidade por falta de sementes.

6.2 Política de incentivos

O sistema baseado em Trusted Peers tem como base de funcionamento um sistema que dá prioridade aos participantes com o estatuto de Trusted Peer. Este procedimento é por si só um sistema de incentivos: é possível aliciar os participantes a permanecer na rede de partilha mais tempo com o objectivo de alcançarem o estatuto de Trusted Peers, e consequentemente transferências mais rápidas.

Um Trusted Peer é um participante cujo comportamento é benéfico para a rede, ou seja, um participante que por norma se mantém ligado em modo de semente para ajudar outros a terminar as suas transferências. O sistema dá prioridade a estes participantes não apenas como forma de os recompensar e incentivar, mas também por conseguir melhorar o desempenho devido à criação de sementes mais rapidamente. Trata-se de um mecanismo com potencial de sucesso pois permite que todas as partes ganhem.

Um utilizador deve ser considerado Trusted Peer se tiver um rácio global (isto é, de todas as suas transferências passadas e presentes) francamente superior a 1. Rácio é a medida de desempenho dada por *bytes* oferecidos a dividir por *bytes* recebidos. Para manter o estatuto, um utilizador tem de manter um comportamento benéfico para a rede, oferecendo blocos a outros, mantendo desta forma o seu rácio acima de 1.

Este sistema de incentivos funciona junto dos utilizadores na medida em que aqueles que têm um comportamento altruísta são recompensados, os que pretendem obter transferências mais rápidas são incentivados a ter um comportamento mais benéfico para a rede, e os restantes utilizadores podem continuar o comportamento habitual, mas contando com uma rede com melhores desempenhos.

6.3 Implementação

Para que seja possível a implementação da solução em sistemas reais seria necessário que diversas condições se verificassem. Em primeiro lugar é necessária a existência de um qualquer mecanismo capaz de atribuir e comprovar o estatuto de Trusted Peer, o que requer a noção de identidade de cada utilizador, que pode ser alcançada através de registos e autenticação. A rede deve ainda ser constituída apenas por clientes que executem esta solução, caso contrário não seria possível dar prioridade aos Trusted Peers.

Atendendo aos requisitos apresentados, aponta-se para a distribuição deste sistema em comunidades privadas, que requerem autenticação e cuja maioria oferece já uma entidade que gere as estatísticas de cada utilizador, entre elas o rácio. Nestas comunidades é necessária autenticação, e os clientes utilizados são muitas vezes proprietários, tal como descrito já na secção 2.6.3. Distribuir o sistema segundo estes parâmetros tornaria a implementação mais fácil, e a aceitação deste tipo de redes pela parte dos utilizadores já está comprovada.

A forma menos restritiva de colocar este sistema de incentivos em prática seria não oferecer sanções aos utilizadores com pior rácio, como a maioria das comunidades fechadas faz. Em vez disso apenas seriam recompensados os utilizadores altruístas com prioridade nas suas transferências.

Este sistema, como qualquer outro, está sujeito a falhas e ataques. Foi no entanto desenhado de forma a possuir alguma robustez, embora nalguns casos sejam necessárias alterações para o tornar seguro face a um ou outro tipo particular de ataque.

Sendo baseado numa rede fechada com cliente privado, a utilização de clientes que recorrem a estratégias com o objectivo de conseguir ludibriar o sistema e conseguir efectuar a transferência de forma mais rápida é posta de parte.

A utilização de identidades falsas também não se coloca devido a uma qualquer forma de certificação efectuada na entidade central que atribui o estatuto de Trusted Peer.

O *free-riding* é combatido pela base do algoritmo, o protocolo FairTorrent, que não oferece mais do que poucos blocos aos *free-riders*, que ficam dependentes maioritariamente das sementes, uma estratégia que não compensa a longo prazo. Por outro lado, se as sementes também privilegiarem os Trusted Peers, os *free-riders* apenas terão acesso à capacidade não utilizada da rede, inviabilizando ainda mais a sua tentativa de ludibriar o sistema.

A entidade central que atribui o estatuto de Trusted Peer é um ponto de falha, que pode sofrer facilmente ataques de *denial of service*, entre outros. O sistema pode no entanto reverter à execução do protocolo FairTorrent original no caso de não ser capaz de validar nenhum participante, contornando assim de forma temporária a falha.

Por último, para obter o estatuto de Trusted Peer é necessário enviar para a entidade central o tráfego efectuado e, partindo do princípio que o código do cliente não pode ser alterado e que as mensagens trocadas entre as entidades são seguras, este comprova que o participante tem de facto um rácio superior a 1. Há no entanto formas de conseguir obter o estatuto sem necessitar de permanecer ligado em modo de semente para outros participantes: o *Sybil Attack*. Um ataque desta natureza consiste na criação de diversas identidades pelo mesmo utilizador,

que são usadas para alcançar um qualquer fim e depois rapidamente descartadas. Neste caso, seria possível criar um qualquer número grande de identidades para fazer transferências de quaisquer conteúdos entre as mesmas, de forma a fazer subir o rácio de uma delas até a mesma atingir o estatuto de Trusted Peer. Esta estratégia pode ser combatida com recurso a regras restritas na criação de identidades (criação por convite com responsabilidade para os emissores dos convites, listas de espera ou outras estratégias).

O cenário e os requisitos descritos para que a utilização da proposta dos Trusted Peers fosse viável são restritivos e pouco populares. Tal pode prejudicar ou até mesmo impossibilitar a sua utilização na prática, ou mesmo interferir na sua popularidade a um nível tal que torne inviável a sua utilização.

Existe no entanto um cenário no qual a utilização desta solução e deste algoritmo nos parece mais viável. Trata-se de um cenário de rede de operador em que o algoritmo seria executado pelas *set top boxes* dos clientes. Usar as mesmas como cliente de um protocolo deste tipo permitiria aos operadores disponibilizar o acesso a conteúdos incluídos na subscrição de base, sem que o operador tivesse que investir em infra-estruturas de distribuição de conteúdo muito dispendiosas. Os utilizadores que não desligassem a sua *set top box* em períodos em que não estavam interessados em conteúdos seriam recompensados por receberem mais depressa os conteúdos quando necessitassem. Como é fácil de avaliar a contribuição de rede de cada utilizador neste ambiente, seria fácil dar *feed-back* sobre o seu estatuto. Mais detalhes acerca deste serviço teriam de ter em medida de conta o conteúdo a distribuir e o publico alvo, assim como a política do operador.

6.4 Conclusões

Este capítulo abordou as vantagens obtidas pelo sistema Trusted Peers, assim como uma possível distribuição de uma implementação em redes reais. Isto levanta problema subjectivos como o funcionamento da política de incentivos e o comportamento dos utilizadores, e problemas concretos, como é o caso da segurança.

No geral espera-se ser possível alcançar resultados semelhantes aos obtidos nos testes por simulação e garantidos pelo sistema de incentivos, caso os utilizadores se comportem da forma esperada, com uma percentagem a alterar o seu comportamento devido aos incentivos, melhorando assim os tempos de transferência de todos. No entanto esta projecção é meramente

teórica, e apenas poderia ser provada por trabalho futuro.

Descrevemos igualmente um cenário de aplicação real que evita os problemas delicados de segurança que a solução requer.

7 . Resumo, conclusões e trabalho futuro

Esta dissertação visa estudar formas de melhorar o algoritmo BitTorrent através de políticas de incentivos que ultrapassam o conceito de troca directa que caracteriza o protocolo original.

Assim, começou-se por dissecar em pormenor o funcionamento do algoritmo BitTorrent, e as razões que o elevam a uma das principais formas de partilha de conteúdo em sistemas baseados em P2P. De seguida abordaram-se várias propostas de melhoramentos do algoritmo. Algumas passaram por melhoramentos específicos para cenários específicos, como o caso do BitMax, outras por estratégias alternativas com o objectivo de mostrar as fragilidades do sistema, como o BitTyrant, outras ainda abordam o problema dos ISPs, que têm de lidar com a grande quantidade de tráfego que o protocolo gera. Foram ainda analisadas propostas que introduziram conceitos novos como o caso do TRIBLER e do 2Fast com os seus Collectors e Helpers, o sistema PACE com a introdução de um sistema baseado em noções económicas de equilíbrio de mercado, e por fim os sistemas PropShare e FairTorrent, com um novo conceito de troca directa.

Após estes primeira fase de análise do trabalho relacionado, foram analisadas as ferramentas a utilizar na experimentação e equacionadas as hipóteses de recorrer a simulação ou a emulação, tendo sido escolhida a primeira via. Depois de seleccionada a ferramenta de trabalho, o simulador GPS-P2P, este foi alvo dos melhoramentos que se revelaram necessários para realizar o trabalho, o que levou a um estudo aprofundado do código do mesmo, e à implementação de algumas das propostas recenseadas no simulador, bem como algumas soluções novas, para serem submetidas a testes.

Para além do protocolo BitTorrent foram submetidas a testes por simulação 4 propostas:

- Protocolo 2Fast
- Helpers colectivos
- Protocolo FairTorrent
- Trusted Peers

A primeira observação confirmada pelos testes foi o impacto que o número de sementes e o comportamento dos utilizadores no final da transferência têm sobre o desempenho da rede de partilha. Confirmou-se assim a importância de incentivar os utilizadores a permanecerem na rede em modo semente.

O protocolo 2Fast foi testado e obtiveram-se bons resultados, tal como já demonstrados pelos autores da proposta. Observou-se existir nesta proposta espaço para a construção de um sistema de incentivos para rentabilizar a largura de banda disponível de alguns utilizadores, que poderiam assim ajudar outros nas suas transferências, o que confirma a viabilidade do sistema TRIBLER.

Baseada no protocolo 2Fast foi explorada a solução designada de Helpers colectivos, que consiste na existência de utilizadores que se juntam à rede para transferir apenas determinada percentagem do ficheiro e permanecer depois ligados na rede, a oferecer a outros aquilo que conseguiram sem pedir nada em troca. Esta solução revelou não cumprir os objectivos, e como tal, estudos mais aprofundados sobre a mesma foram abandonados.

Os testes efectuados ao protocolo FairTorrent vieram comprovar os bons resultados publicados pelos autores, e demonstraram existir um grande potencial nesta proposta devido aos bons desempenhos obtidos.

A solução com o nome de Trusted Peers surgiu assim no seguimento dos testes com o protocolo FairTorrent. Pegando no sistema implementado e introduzindo ligeiras alterações de forma a ser possível dar prioridade a utilizadores que se sabe com relativa confiança que irão permanecer ligados em modo semente quando terminarem a transferência, é possível aumentar o desempenho do sistema. Esta solução não se limita a oferecer melhores resultados para os utilizadores a quem é dada prioridade na rede, chamados de Trusted Peers, os restantes também obtém tempos de transferência menores. Foram registadas melhorias de tempo entre 10% a 20% para os Trusted Peers, e de cerca de 6% para os restantes. Trata-se portanto de uma solução em que ganham todos os participantes.

O sistema de incentivos proposto tem por base a análise do comportamento dos utilizadores actuais de redes baseadas em sistemas BitTorrent, onde se observa que a maioria desses utilizadores tem rácios na ordem dos 0.5 a 0.6, e que 78% da carga de transferência é normalmente acarretada por apenas 10% dos utilizadores, geralmente as sementes iniciais e uns poucos utilizadores que mantém um comportamento altruísta permanecendo na rede para ajudar outros a completar as suas transferências. Assim a solução Trusted Peers vem recompensar estes poucos utilizadores sobre os quais assenta a maioria da carga das transferências do sistema, aliciando a que mais utilizadores se juntem a este grupo, melhorando o desempenho não só para estes mas também para os restantes utilizadores, que vêem o seu tempo de transferência diminuído.

Discutem-se igualmente as modalidades de uma possível implementação deste sistema e o que isso acarretaria em termos de restrições e segurança. Adicionalmente é apresentado um

possível ambiente real para a distribuição deste sistema: uma rede de distribuição de conteúdo através de *set top boxes* de clientes de uma operadora, que assim distribuía conteúdo sem necessitar de investir em infra-estruturas, e recompensava com transferências mais rápidas os clientes que ajudavam na distribuição mantendo as suas *set top boxes* ligadas e a partilhar conteúdo.

A solução Trusted Peers, que é a principal contribuição inovadora deste trabalho, mostra ser promissora e está assim aberta a trabalho futuro. Seria importante efectuar o estudo de formas alternativas de dar prioridade aos nós. Por exemplo em vez do uso de uma constante booleana (ser ou não ser Trusted Peer) poderia ser usado o valor do rácio no cálculo da prioridade, dando mais prioridade a utilizadores com maiores rácios, aumentando assim a probabilidade de escolher os participantes correctos e oferecendo um sistema mais justo. A realização de testes mais exaustivos e baseados em redes assimétricas, no Planet-Lab ou mesmo em redes reais, seriam um passo importante no estudo desta solução.

Além de um estudo mais aprofundado da solução Trusted Peers e de uma eventual implementação e distribuição em ambiente real, seria também interessante como trabalho futuro introduzir diversas melhorias no simulador usado, como a capacidade de simular redes com linhas assimétricas. O simulador GPS-P2P revelou-se uma ferramenta muito útil no decorrer deste trabalho, apesar das limitações reveladas e das intervenções que foi alvo.

É também claro que o trabalho do ponto de vista formal está incompleto. Por um lado poderia ser realizada uma análise do algoritmo, partindo das análises apresentadas em [16] sobre o FairTorrent e completando-as com o mecanismo Trusted Peers. Por outro lado é crucial realizar um estudo mais aprofundado da problemática de segurança que em larga medida foi ignorada.

Assim, este trabalho estudou o algoritmo BitTorrent e algumas propostas de melhoramentos, testou-as em ambiente de simulação, bem como duas soluções originais desenhadas a partir das estudadas, de onde sobressai a solução Trusted Peers, um sistema capaz de incentivar e recompensar os utilizadores a permanecerem como sementes no sistema, oferecendo uma rede com melhor desempenho para todos.

Bibliografia

[1] Nazareno Andrade, Miranda Mowbray, Aliandro Lima, Gustavo Wagner, and Matei Ripeanu. Influences on cooperation in bittorrent communities. In *2005 ACM SIGCOMM Workshop on Economics of peer-to-peer systems*, pages 111–115. ACM, 2005.

[2] Christina Aperjis, Michael J. Freedman, and Ramesh Johari. Peer-assisted content distribution with prices. In *Proceedings of the 2008 ACM CoNEXT Conference*. ACM, 2008.

[3] Ruchir Bindal, Pei Cao, William Chan, Jan Medval, George Suwala, Tony Bates, and Amy Zhang. Improving traffic locality in bittorrent via biased neighbor selection. In *Proceedings of the 26th IEEE International Conference on Distributed Computing Systems*. IEEE Computer Society, 2006.

[4] Bram Cohen. Incentives build robustness in bittorrent. In *1st Workshop on Economics of Peer-to-Peer Systems*, 2003.

[5] Pawel Garbacki, Dick H. J. Epema, and Maarten van Steen. An amortized tit-for-tat protocol for exchanging bandwidth instead of content in p2p networks. In *Proceedings of the First International Conference on Self-Adaptive and Self-Organizing Systems*, pages 119–128. IEEE Computer Society, 2007.

[6] Pawel Garbacki, Alexandru Iosup, Dick Epema, and Maarten van Steen. 2fast: Collaborative downloads in p2p networks. In *Proceedings of the Sixth IEEE International Conference on Peer-to-Peer Computing*, pages 23–30. IEEE Computer Society, 2006.

[7] M. Izal, G. Urvoy-Keller, E. W. Biersack, P. A. Felber, A. Al Hamra, and L. Garcés-Erice. Dissecting bittorrent: Five months in a torrent's lifetime. In *Proceedings of Passive and Active Measurements 2004*. Springer, 2005.

[8] Nikolaos Laoutaris, Damiano Carra, and Pietro Michiardi. Uplink allocation beyond choke/unchoke. In *Proceedings of the 2008 ACM CoNEXT Conference*. ACM, 2008.

[9] Dave Levin, Katrina LaCurts, Neil Spring, and Bobby Bhattacharjee. Bittorrent is an auction: Analyzing and improving bittorrent's incentives. In *Proceedings of the ACM Conference on Applications, Technologies, Architectures, and Protocolos for Computer Communications (SIGCOMM '08)*. ACM, 2008.

[10] Michael Piatek, Tomas Isdal, Thomas Anderson, Arvind Krishnamurthy, and Arun Venkataramani. Do incentives build robustness in bittorrent? In *NSDI '07: 4th USENIX Symposium on Networked Systems Design & Implementation*, 2007.

[11] PlanetLab. Planetlab. `http://www.planet-lab.org/`, As of July 2009.

[12] J. A. Pouwelse, P. Garbacki, D. H. J. Epema, and H.J. Sips. The bittorrent p2p file-sharing system: Measurements and analysis. In *Proceedings of the IPTPS '05*. Springer, 2005.

[13] J. A. Pouwelse, P. Garbacki, J. Wang, A. Baker, J. Yang, A. Iosup, D. H. J. Epema, M. Reinders, and M. R. van Steen. Tribler: a social-based peer-to-peer system. *Concurrency and Computation: Practice & Experience*, 20(2):127–138, 2008.

[14] Matei Ripeanu, Miranda Mowbray, Nazareno Andrade, and Aliandro Lima. Gifting technologies: A bittorrent case study. In *First Monday, vol. 11, issue 11*. http://firstmonday.org, 2006.

[15] UCSD Computer Science. Modelnet. `https://modelnet.sysnet.ucsd.edu/`, As of July 2009.

[16] Alex Sherman, Jason Nieh, and Clifford Stein. Fairtorrent: Bringing fairness to peer-to-peer systems. In *Proceedings of the 2009 ACM CoNEXT Conference*, pages 133–144. ACM, 2009.

[17] Wikipedia. Vuse. `http://en.wikipedia.org/wiki/Azureus_Vuze`, As of July 2009.

[18] Weishuai Yang and Nael Abu-Ghazaleh. Gps: A general peer-to-peer simulator and its use for modeling bittorrent. In *Proceedings of the 13th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, pages 425–434. IEEE Computer Society, 2005.

[19] Weishuai Yang and Nael Abu-Ghazaleh. Gps - general purpose simulator for p2p network. `http://www.cs.binghamton.edu/~wyang/gps/`, As of July 2009.